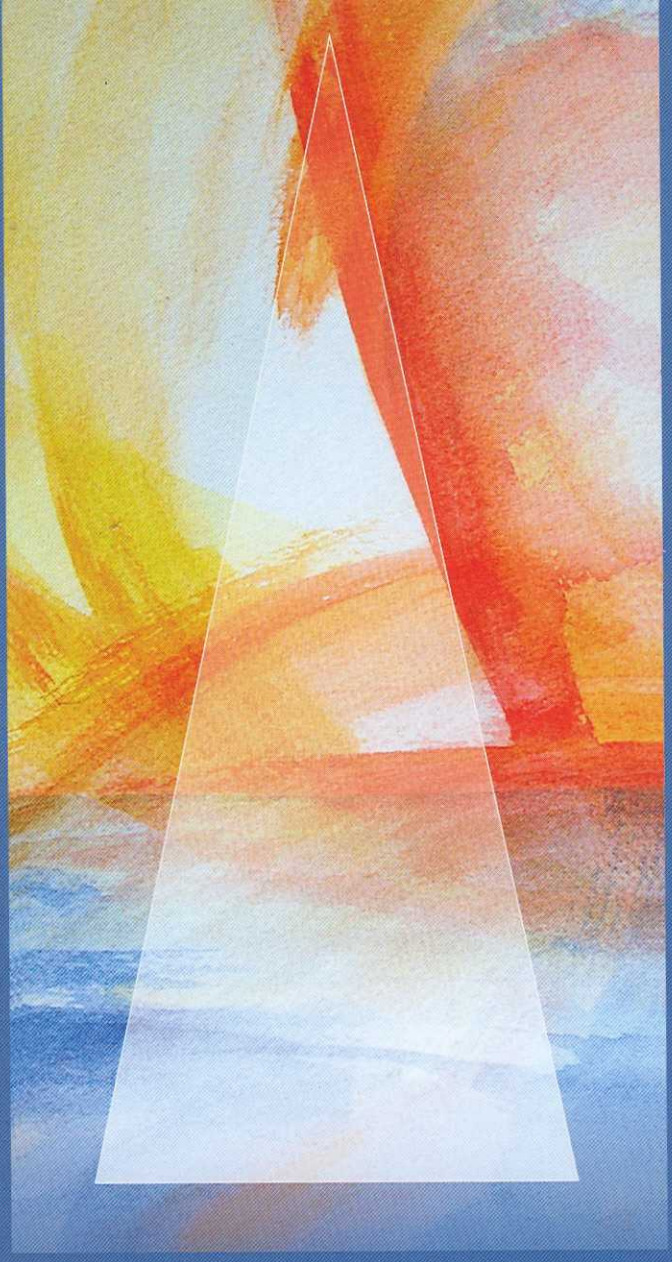


IMREH BALÁZS – IMREH CSANÁD

KOMBINATORIKUS OPTIMALIZÁLÁS

EGYETEMI TANKÖNYV

KOMBINATORIKUS OPTIMALIZÁLÁS

Imreh Balázs és Imreh Csanád
Szegedi Tudományegyetem
Informatikai Tanszékcsoport

2005. november 15.

Tartalomjegyzék

Előszó

Az operációkutatás fejlődése a vizsgált modellek számának gyors növekedését, és a modellekhez kapcsolódó elméleti háttér mély és alapos megismerését eredményezte. Ennek következtében az egész diszciplina nagymértékben kiterjedt, és elkezdődött bizonyos részterületek szeparálódása, önállósodása. Ezek egyikének tekinthető a kombinatorikus optimalizálás témaköre, melynek gyakorlati szempontból egyre nagyobb a jelentősége.

A tekintett terület szeparálódásához az a felismerés vezetett, hogy bizonyos egészértékű problémák megoldásában felhasználhatók a feladatok kombinatorikus tulajdonságai, amelyek esetenként igen hatékony megoldási eljárások felépítését teszik lehetővé. Számos ilyen eljárás kidolgozása után megkezdődött ezek szintetizálása, általános kombinatorikus technikák, módszerek kidolgozása. Az elért eredmények gyakorlati alkalmazását nagyban elősegítette a gyorsan fejlődő számítástechnika. Mindezek azt eredményezték, hogy az operációkutatáson belül kialakult egy igen jelentős részterület, a kombinatorikus optimalizálás témaköre. A terület eredményeit tartalmazó, jó összefoglaló munkák a [38], [45], [54], [102], [103], [112], [122], [147] és [168] könyvek.

Tekintettel a témakör számos fontos gyakorlati alkalmazására, a kombinatorikus optimalizálás részben más tantárgyak részeként részben önálló tantárgyként bekerült a felsőoktatásba és várható a diszciplina további térnyerése az oktatásban. Ezekhez kíván segítséget adni a jelen munka, amelyben betekintést szeretnénk nyújtani ezen terület bizonyos részeibe.

Rövid gráfelméleti bevezetés után elsőként a párosítási feladatokat vizsgáljuk, majd ismertetjük a hálózatokra vonatkozó legrövidebb utak problémáját, és néhány, a probléma megoldására szolgáló, ma már klasszikusnak tekinthető eljárást. Ezt követően a hálózati folyamok problémakörét és a megoldást biztosító Ford-Fulkerson féle címkézési eljárást tárgyaljuk. A következő rész tartalmazza a kombinatorikus optimalizálás egyik leggyakrabban alkalmazott technikájának, a korlátozás és szétválasztás módszerének tárgyalását. Ezt a korlátozás és szétválasztás módszerének több, szép alkalmazása követi. Az első igen szép alkalmazás a hátizsák feladat megoldására szolgáló eljárás, ezt követi a korlátos egészértékű lineáris programozási feladatok megoldására szolgáló eljárás. Ezek után a kombinatorikus opti-

malizálás egy nevezetes feladatát, az utazó ügynök problémáját vizsgáljuk. A probléma megoldására alkalmazzuk a korlátozás és szétválasztás módszerét, majd két külön fejezetben az utazó ügynök problémára és az azzal rokon feladatokra szuboptimális megoldást szolgáltató heurisztikákat ismertetünk. Az utóbbi két fejezet egyben betekintést nyújt az egyre nagyobb teret nyerő heurisztikus eljárások elméletébe is. Egy másik nevezetes modell, a halmazlefedési probléma vizsgálatával folytatódik a tárgyalás. A probléma megoldására felépítünk egy, a korlátozás és szétválasztás módszerén alapuló eljárást, majd megadunk egy viszonylag jó heurisztikus eljárást is. Ezek után a diszkrét determinisztikus kiszolgáló telepítési problémákat érintjük, nevezetesen a medián, center és a kvadratikus hozzárendelési feladatokat tárgyaljuk. A kiszolgálási feladatok után az ütemezési problémák legegyszerűbb eseteivel foglalkozunk. Végül az approximációs sémákat érintjük. Definiáljuk a polinomiális approximációs sémák és teljesen polinomiális approximációs sémák fogalmát, majd példákat mutatunk ilyen sémákra az ütemezési problémák köréből.

Miután a tekintett témakörök szinte mindegyike igen nagy, ezért ezek tárgyalása csak bevezető jellegű, és sok esetben a kapcsolódó legegyszerűbb modellekre adunk meg megoldási eljárásokat. Az egyes területek iránt érdeklődők az irodalomjegyzékben iránymutatást kaphatnak a megfelelő könyvekhez, cikkekhez.

A tárgyalás során rendre feltételezzük bizonyos alapismeretek meglétét. A felépítésre kerülő eljárások helyességének igazolásától terjedelmi okokból esetenként eltekintünk, de az eljárásokat rendre egyszerű numerikus példákon keresztül demonstráljuk.

Végezetül szeretnénk köszönetet mondani Blázsik Zoltánnak és Holló Csabának, a jelen munka szakmai lektorainak, akik igen gondos és alapos lektori munkájukkal, valamint hasznos észrevételeikkel segítették a kézirat véglege formájának elkészítését. Mind az említett kollegák, mind jómagunk nagyon reméljük, hogy az elkészült anyag jó hasznára válik valamennyi hallgatónak, aki kombinatorikus optimalizálást tanul, továbbá azoknak is akik csak érdeklődnek a témakör iránt. Amennyiben ezt sikerül elérnünk, akkor a jelen munka hozzájárulás lehet ahhoz, hogy ez a gyorsan fejlődő és fontos diszciplina hazai körökben is ismertebbé váljon, alkalmazásaival a mindennapi élet javulását szolgálhassa.

Szeged, 2005. március 18.

Imreh Balázs és Imreh Csanád

1. Bevezetés

A kombinatorikában általában egy véges halmaz különböző tulajdonságait vizsgálták, és főleg a leszámlálás és a létezés problémájával foglalkoztak. Tipikus kombinatorikus kérdések voltak, hogy valamilyen objektumnak létezik-e adott tulajdonságú elrendezése vagy hány megfelelő elrendezése van. A kombinatorikus optimalizálás egy új kérdést vizsgál, nem az elrendezések száma érdekli, hanem azok közül a legjobb. Az ilyen jellegű problémák formálisan, többnyire egy véges halmazon értelmezett függvény szélsőértékének kereséseként írhatók le. Annak ellenére, hogy ilyen típusú kérdések gyakorlati szempontból már régóta rendkívül érdekesek, a tudományág csak az utóbbi időben indult igazán fejlődésnek. Tulajdonképpen a létét a modern számítógépek megjelenésének köszönheti, hiszen a kombinatorikus optimalizálási problémák megoldására kidolgozott algoritmusok igen sok számítást igényelnek, így számítógép nélkül gyakorlatban is előforduló problémákra nem igazán alkalmazhatók. Továbbá a számítógépek tervezése és a számítógépekkel kapcsolatos egyéb kérdések is számos olyan újabb problémát vetettek fel, amelyek a kombinatorikus optimalizálás témaköréhez tartoznak.

Mint már említettük egy kombinatorikus optimalizálási probléma többnyire egy diszkrét függvény optimumának meghatározása. Egy ilyen probléma megoldásának egy olyan eljárást tekintünk, amely minden feladatra kiszámítja az optimális megoldást. Másrészt minden kombinatorikus optimalizálási feladatnak csak véges sok megoldása van, így egy lehetséges eljárás az, amely egyenként megvizsgálja a feladat lehetséges megoldásait és kiválasztja azok közül a legjobbat. Ezzel az eljárással két probléma is van. Egyrészt nem nyilvánvaló, miként lehet sorba venni az összes lehetséges megoldást (például lehet a lehetséges megoldások halmaza egy gráf azon részgráfjainak halmaza, amelyek valamilyen tulajdonságot kielégítenek). A másik probléma ezzel az eljárással, hogy általában gyakorlatilag nem működik. A lehetséges megoldások halmaza már viszonylag kicsi feladatokra is igen nagy lehet, így ezek mindegyikének megvizsgálása még a jelenleg ismert leggyorsabb számítógépeken is évezredekig tartana. Tehát gyakorlati szempontból nem tekinthetünk minden algoritmust a probléma hatékony megoldásának, csak azokat, amelyeknek nem tart sokáig a feladatok megoldása. Az algoritmusok gyorsaságát az algoritmus által végrehajtott elemi számítási lépések számával mérjük.

Az algoritmusok gyorsaságának egy pontos definíciója adható meg az

algoritmusok Turing gépeken alapuló tárgyalásában, (ld. [68], [154]). Ez a formális, precíz definíció túlmutat a jelen munka keretein, és nem is célunk ezen téma tárgyalása. Itt a továbbiakban az algoritmusok bonyolultságára, gyorsaságára adott becslésekben ugyanazt az egyszerű modellt használjuk, amelyet E. L. Lawler használt a [122] munkában. A modellben lényegében minden egyszerű utasítást (pl. egész aritmetikájú műveletek, numerikus összehasonlítások) egyetlen elemi számítási lépésnek tekintünk. Nyilván a valóságban egy-egy ilyen lépés végrehajtási ideje függ a használt értékek nagyságától, de a jegyzet céljainak megfelel ez a leegyszerűsített modell is.

Egy konkrét feladat megoldásához szükséges elemi számítási lépések száma nyilvánvalóan függ a tekintett feladattól is. (Pl. egy nagyobb gráf csúcsainak megvizsgálásához több lépés kell, mint egy kisebb gráf esetén). Tehát azokat az algoritmusokat érdemes egy probléma hatékony megoldó algoritmusainak tekintenünk, amelyekre teljesül, hogy minden egyes feladat esetén a feladat megoldásához szükséges elemi lépések számára egy általános, csak a feladat nagyságától függő, jó korlát adható. Általában a kombinatorikus optimalizálásban egy algoritmust akkor tekintünk hatékonynak, ha *polinomiális időigényű*, azaz ha az algoritmus által megkívánt elemi lépések száma a feladat méretének egy polinomiális függvényével korlátozható.

Ez a definíció két nyilvánvaló kérdést vet fel. Az első az, hogy miként értelmezzük egy feladat méretét. A második pedig az, hogy miért éppen a polinomiális időigényű eljárásokat tekintjük hatékonynak. Egy feladat méretén általában a lekódolásához szükséges bitek számát értjük. Ennek ellenére a szakirodalomban gyakran az egyszerűség érdekében más könnyebben kezelhető paramétert vizsgálnak, és ezt a szemléletet követjük a jelen tárgyalásban is. Példaként felhozhatjuk, hogy egy súlyozott gráf lekódolásának mérete az élekre írt számok kettes alapú logaritmusainak összege plusz az élek kódolása, mégis a feladat méretén általában a gráf csúcsainak számát értjük. Ezzel a feladat méretét alulról becsüljük, így ha az eljárás polinomiális lesz a csúcsok számában, akkor a kódoláshoz szükséges bitek számában is. Fontos megjegyeznünk, hogy az egyszerűség érdekében a méretre tett feltevéseink nem vezethetnek oda, hogy egy algoritmust polinomiális időigényűnek nyilvánítsunk, amennyiben az nem az.

A következő kérdés, hogy miért éppen a polinomiális időigényű eljárásokat tekintjük hatékonynak. Ismert, hogy a polinomfüggvények sokkal lassabban növekszenek, mint az exponenciális vagy faktoriális függvények. Tehát ha egy algoritmusra a szükséges lépések száma $100n$ egy másikra pedig 2^n, akkor a 17-nél kisebb n-ekre a második algoritmus a hatékonyabb, a 17-nél a nagyobb n-ekre pedig az első. Viszont kis n-ekre mindkét algoritmus gyorsan végrehajható, $n = 100$-ra pedig a polinomiális algoritmus még mindig

gyorsan megoldja a problémát, ellenben az exponenciális algoritmussal való megoldása rendkívül hosszú ideig tartana.

Természetesen ez nem jelenti azt, hogy a polinomiális algoritmusok a gyakorlatban mindig jobbak az exponenciális algoritmusoknál. Például egy n^{10000} időigényű algoritmus semmivel sem használhatóbb gyakorlati problémák esetén egy exponenciális időigényű eljárásnál. Itt érdemes megjegyeznünk, hogy az eddigi tapasztalatok szerint a polinomiális algoritmusok majdnem mindig hatékonyak, és a korlátozó polinom foka általában 5 alatt van. Másrészt azt is el kell mondanunk, hogy vannak olyan exponenciális algoritmusok, amelyek a legtöbb feladat esetén rendkivül gyorsan megadják az optimális megoldást, és a feladatok csak egy nagyon szűk osztályán szükséges exponenciálisan sok lépés. Ilyen eljárás a lineáris programozási feladatot megoldó szimplex algoritmus. Erre a problémára van polinomiális algoritmus is, de számítógépen lényegesen nehezebben megvalósítható, mint a szimplex eljárás és csak nagyon nagy feladatok esetén hatékonyabb.

Természetesen nem egyformán jó egy n és egy n^2 időigényű algoritmus. Annak érdekében, hogy a polinomiális algoritmusok időigényeit is összehasonlíthassuk, szükségünk van a következő jelölésekre. Legyen f és g, két a természetes számokról a természetes számokra képező függvény. Az $f(n) = O(g(n))$ ($f(n)$ *egyenlő nagyordó* $g(n)$) jelölés azt jelenti, hogy léteznek olyan c és n_0 pozitív egész számok, amelyekre teljesül, hogy minden $n \geq n_0$ számra $f(n) \leq c \cdot g(n)$. Szemléletesen ez a jelölés azt jelenti, hogy f nem nő gyorsabban mint g. Az $f(n) = \Omega(g(n))$ jelölést akkor használjuk, ha $g(n) = O(f(n))$, az $f(n) = \Theta(g(n))$ jelölést pedig akkor, ha $f(n) = O(g(n))$ és $f(n) = \Omega(g(n))$ egyaránt teljesül. Egyszerűen látható, hogy tetszőleges k-adfokú polinom $O(n^k)$ nagyságrendű, és az is, hogy minden $c > 1$ és $k > 1$ egész esetén $c^n = \Omega(n^k)$.

Fontos megjegyeznünk, hogy nem minden feladat megoldására létezik polinomiális algoritmus. Annak vizsgálatával, hogy az egyes feladatok megoldhatósága miként viszonyul egymáshoz egy külön terület, a *bonyolultságelmélet* foglalkozik. A jelen munkában nem kívánunk ezzel a területtel foglalkozni (az olvasó részletes felépítést találhat a [68], [150] munkákban). Annyit azért fontos megjegyeznünk, hogy van a feladatoknak egy olyan osztálya, az *NP-teljes feladatok*, amelyek közül bármelyikre polinomiális algoritmust találva, az összes többi is polinomiálisan megoldható lenne. Mivel ezen feladatok egyikére sem sikerült eddig polinomiális algoritmust találni, ezért igen elterjedt (de nem bizonyított) feltételezés, hogy ezen feladatok nem oldhatók meg polinomiális időben. Ilyen feladatok esetén igyekszünk exponenciális, de gyakorlatban mégis hatékony eljárásokat kidolgozni.

2. Gráfok, hálózatok

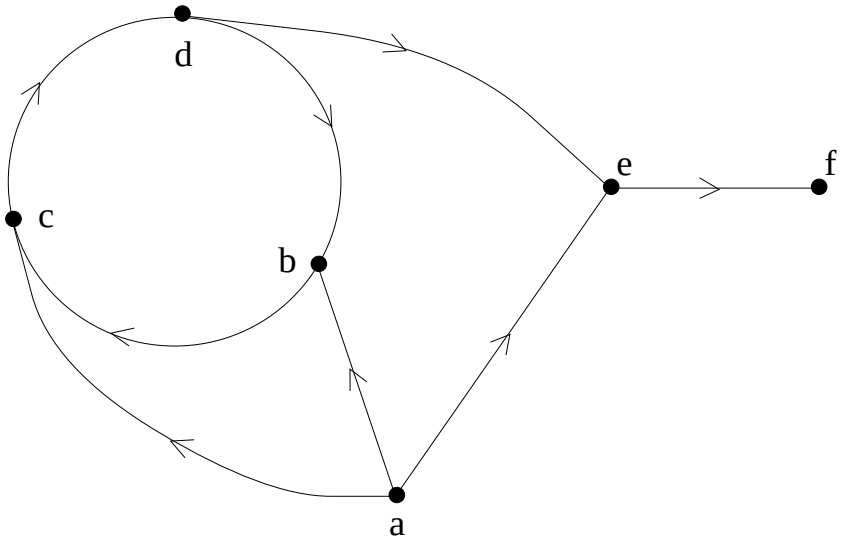
A jelen fejezetben felidézzük a véges irányított gráfok fogalmát, azok néhány alapvető tulajdonságát, bizonyos reprezentációjukat. Ezek után véges irányítatlan gráfokkal foglalkozunk, majd a fejezet végén bevezetjük a hálózat fogalmát, és annak néhány reprezentációját mind az irányított, mind az irányítatlan gráfokra.

Véges irányított gráfon egy $\mathcal{G} = (V, E)$ párt értünk, ahol V véges, nemüres halmaz, a *gráf csúcsainak* vagy *szögpontjainak* a halmaza, $E \subseteq V \times V \setminus I_V$ a *gráf éleinek* a halmaza, ahol I_V a V halmaz feletti egyenlőség relációját jelöli. A fenti definíció azt jelenti, hogy a (v, v) $(v \in V)$ típusú éleket, amiket *loop éleknek* nevezünk, nem engedjük meg, azaz csak olyan gráfokat tekintünk, amelyek nem tartalmaznak loop élet.

Azt mondjuk, hogy a $\mathcal{G}$ gráf $u \in V$ csúcsából él vezet a $v \in V$ csúcsba, ha $(u, v) \in E$, továbbá ez esetben a v csúcsot az u csúcs *leszármazottjának*, u-t pedig a v csúcs *ősének* nevezzük. A gráf egy olyan pontját, amelynek nincsen egyetlen őse sem (feltéve, hogy van ilyen csúcs) *forrásnak*, míg egy olyan csúcsot, amelynek nincsen egyetlen leszármazottja sem, *nyelőnek* nevezzük. Ha $v_1, \ldots, v_k$ $(k \geq 2)$ a gráf páronként különböző csúcsai, és $(v_i, v_{i+1}) \in E$, $i = 1, \ldots, k-1$, akkor a $(v_1, v_2), \ldots, (v_{k-1}, v_k)$ élek sorozatát *irányított útnak* nevezzük, melynek v_1 a *kezdőpontja* és v_k a *végpontja*. Ha a (v_k, v_1) él is eleme az E élhalmaznak, akkor a fenti élsorozatot ezzel az éllel kiegészítve, a kapott élsorozatot *irányított körútnak* nevezzük, és azt modjuk, hogy $\mathcal{G}$ tartalmaz irányított kört. Miután az utat úgy definiáltuk, hogy a benne szereplő csúcsok páronként különbözők, ezért a körutat nem tekintjük útnak, hanem külön fogalomként kezeljük. A $\mathcal{G}$ irányított gráfot *körmentesnek* nevezzük, ha nem tartalmaz irányított körutat.

Mivel a továbbiakban főként irányított gráfokkal fogunk dolgozni, ezért az egyszerűbb szóhasználat érdekében az irányított út és irányított körút esetében elhagyjuk az "irányított" jelzőt, azaz úton és körúton irányított gráfok esetében mindig irányított utat és körutat értünk. Amennyiben ettől eltérünk, azt külön fogjuk hangsúlyozni.

Egy adott $\mathcal{G}$ gráf többféle, egymással ekvivalens formában adható meg, ezeket a gráf *reprezentációinak* nevezzük. A legszemléletesebb reprezentáció, amikor a gráfot egy ábrával adjuk meg, ezt *grafikus reprezentációnak* nevezzük.



2.1. ábra. A 2.1 példa $\mathcal{G}$ gráfjának grafikus reprezentációja.

Grafikus reprezentáció

Rajzoljuk fel egy síkra a gráf minden csúcsát, feltüntetve a csúcs nevét is, és kössük össze a gráf a, b csúcspárját egy irányított vonallal, ha $(a, b) \in E$. Ezt hajtsuk végre a gráf minden a, b csúcspárjára. Nyilvánvaló, hogy $\mathcal{G} = (V, E)$ egyértelműen meghatározza az ábrát abban az értelemben, hogy milyen pontok szerepelnek rajta és ezek közül melyek vannak irányított vonalakkal összekötve. Továbbá a megfordítás is igaz. Egy ilyen ábrához egyértelműen hozzárendelhető egy irányított $\mathcal{G} = (V, E)$ gráf.

A fentiek illusztrálására tekintsük az alábbi példát.

2.1. példa. Legyen $\mathcal{G} = (V, E)$, ahol $V = \{a, b, c, d, e, f\}$, és az élek halmaza, $E = \{(a, b), (a, c), (a, e), (b, c), (c, d), (d, b), (d, e), (e, f)\}$. A $\mathcal{G}$ gráf grafikus reprezentációját adja meg a 2.1. ábra.

Vegyük észre, hogy a fenti gráfban az a csúcs forrás, az f csúcs pedig nyelő, továbbá a gráf tartalmaz körutat, nevezetesen a $(b, c), (c, d), (d, b)$ élek egy körutat alkotnak.

Egy másik, gyakorlati szempontból nagyon hasznos reprezentációja a gráfoknak, amikor bináris mátrixokkal írjuk le őket, amit mátrix-reprezentációnak szokásos nevezni.

Gráfok mátrix-reprezentációja

Tegyük fel, hogy a gráf n számú csúcspontot tartalmaz. Elsőként vegyük a gráf csúcsainak egy lineáris rendezését. Ezek után képezzük a következő (d_{ij}) $n \times n$-es bináris mátrixot. Legyen $d_{ij} = 1$, ha az i-edik csúcspontból vezet él a j-edik csúcspontba, ahol a csúcsok sorszámozása a tekintett rendezés szerint történik; ha nincs ilyen él, akkor legyen $d_{ij} = 0$. A (d_{ij}) mátrixot a tekintett gráf *mátrix-reprezentációjának* nevezzük. Nyilvánvaló, hogy adott rendezés mellett a (d_{ij}) mátrix egyételműen meghatározott.

A 2.1. példa esetében az alfabetikus rendezés mellett az alábbi mátrix-reprezentációt kapjuk, ahol a sorok mellett és az oszlopok felett a megfelelő csúcsokat is feltüntettük.

$$\begin{pmatrix} & a & b & c & d & e & f \\ a & 0 & 1 & 1 & 0 & 1 & 0 \\ b & 0 & 0 & 1 & 0 & 0 & 0 \\ c & 0 & 0 & 0 & 1 & 0 & 0 \\ d & 0 & 1 & 0 & 0 & 1 & 0 \\ e & 0 & 0 & 0 & 0 & 0 & 1 \\ f & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

A reprezentációs mátrix definíciójából közvetlenül adódnak az alábbi észrevételek.

(1) A mátrix i-edik sorában levő egyesek összege pontosan azt adja meg, hogy az i-edik csúcsból hány másik csúcsba vezet él. Ezt a számot szokásos az i-edik csúcs *kifokának* nevezni. Speciálisan, a sorösszeg akkor és csak akkor 0, ha az illető csúcsból nem vezet semelyik más csúcsba él, így az illető csúcs nyelő. (Például a 2.1. példánál az f csúcsra vonatkozó sorösszeg 0, így az f csúcs a $\mathcal{G}$ gráf nyelője, amint azt már említettük.)

(2) A mátrix j-edik oszlopában lévő egyesek összege pontosan azt adja meg, hogy hány különböző csúcsból vezet él a j-edik csúcsba. Ezt a számot a j-edik csúcs *befokának* nevezzük. Speciálisan, az oszlopösszeg akkor és csak akkor 0, ha nem vezet él a j-edik csúcsba, azaz az illető csúcs forrás. (Például a 2.1. példában az a csúcsra vonatkozó oszlopösszeg 0-val egyenlő, tehát az a csúcs a $\mathcal{G}$ gráf forrása.)

Szükségünk lesz a továbbiakban a részgráf fogalmára. A $\mathcal{G}' = (V', E')$ gráfot a $\mathcal{G} = (V, E)$ gráf *részgráfjának* nevezzük, ha $V' \subseteq V$ és $E' \subseteq E$ teljesül. Megjegyezzük, hogy mivel $\mathcal{G}'$ maga is gráf, ezért $V' \neq \emptyset$ és

$E' \subseteq V' \times V' \setminus I_{V'}$ szintén teljesülnek. Nyilvánvaló, hogy amennyiben a $\mathcal{G}$ gráfból törlünk bizonyos éleket vagy bizonyos pontokat az illető pontokba mutató illetve az azokból kivezető élekkel együtt, akkor egy részgráfhoz jutunk. A megfordítás is igaz, ugyanis $\mathcal{G}$ minden részgráfja előállítható ilyen törlésekkel.

A fentiek alkalmazásaként igazoljuk az alábbi állítást, amely a későbbiek során majd felhasználásra kerül.

2.1. segédtétel. *A $\mathcal{G} = (V, E)$ gráf ($|V| = n$) akkor és csak akkor körmentes, ha csúcsai sorszámozhatók az $1, \ldots, n$, természetes számokkal úgy, hogy minden $i \neq j \in \{1, \ldots, n\}$ párra $(v_i, v_j) \in E$ fennállásából $i < j$ következik.*

Bizonyítás. Elsőként tegyük fel, hogy $\mathcal{G}$ csúcspontjai sorszámozhatók a kívánt módon, azaz teljesül a segédtétel feltétele. Jelölje a sorszámozott csúcsokat $v_1, \ldots, v_n$. Állítjuk, hogy $\mathcal{G}$ nem tartalmaz irányított körutat. Az állítást indirekt bizonyítjuk. Ehhez tegyük fel, hogy $\mathcal{G}$ tartalmaz körutat. Jelölje ennek csúcsait $v_{i_1}, \ldots, v_{i_k}$. Akkor feltételünk szerint $i_1 < i_2 < \ldots < i_k < i_1$, ami ellentmondás. Következésképpen $\mathcal{G}$ nem tartalmazhat körutat.

Most tegyük fel, hogy $\mathcal{G}$ nem tartalmaz körutat. Megmutatjuk, hogy ebben az esetben $\mathcal{G}$ csúcsai sorszámozhatók a kívánt módon. Ezt konstruktívan igazoljuk úgy, hogy megadunk egy olyan eljárást, mely elvégzi a csúcsok sorszámozását.

Mivel $\mathcal{G}$ nem tartalmaz körutat, ezért van legalább egy forrás $\mathcal{G}$-ben. Valóban, ellenkező esetben minden pontnak lenne legalább egy őse, de akkor bármelyik csúcsból kiindulva tudnánk képezni csúcsok egy sorozatát úgy, hogy a sorozat minden csúcspontjához vennénk az illető csúcs őseinek valamelyikét. Mivel minden csúcsnak van legalább egy őse, ezért a sorozat tagjainak képzése vég nélkül folytatható. Másrészt $|V| = n$, így legfeljebb $n - 1$ lépés után a sorozatnak egy olyan tagját kapnánk amely már korábban előfordult a sorozatban. Nyilvánvalóan, a sorozat két azonos tagja közötti csúcsokhoz tartozó élek egy körutat alkotnak, ami ellentmondás. Következésképpen $\mathcal{G}$ tartalmaz legalább egy forrást.

Most vegyük $\mathcal{G}$ egy forrását, jelölje ezt v, és adjuk v-nek az 1 sorszámot. Majd töröljük v-t és a v-ből kiinduló éleket a $\mathcal{G}$ gráfból, és az előálló részgráfot jelölje $\mathcal{G}_1$. Mivel $\mathcal{G}_1$ részgráfja $\mathcal{G}$-nek és $\mathcal{G}$ körmentes irányított gráf, ezért nyilvánvalóan $\mathcal{G}_1$ is körmentes irányított gráf. Ekkor $\mathcal{G}_1$ ismét tartalmaz forrást, ezek közül bármelyiknek adhatjuk a 2 sorszámot, majd ismét töröljük a forrást a belőle kivezető élekkel együtt, és a törlés után előálló részgráfot $\mathcal{G}_2$-vel jelölve, folytatjuk az eljárást. Miután minden lépésben eggyel csökken

a csúcspontok száma, $\mathcal{G}_{n-1}$ már csak egyetlen csúcsot fog tartalmazni, aminek adjuk az n sorszámot.

Mivel minden iterációs lépésben az aktuális részgráfból egy forrást hagyunk el, ezért az illető forrásból csak olyan csúcsokba visznek élek, amelyek később kapnak sorszámot, így sorszámuk nagyobb lesz, mint a forrás sorszáma, amivel adódik az állítás.

Az 2.1. segédtétel bizonyításában megadott eljárás nagyon hatékonyan végrehajtható mátrix-reprezentáció esetén. Amint azt a (2) pont alatt említettük, a j-edik oszlopban szereplő elemek összege akkor és csak akkor 0, ha a j-edik oszlophoz tartozó csúcs forrás. Jelölje a reprezentációs mátrix oszlopösszegeiből álló vektort $\mathbf{r}_0$. Akkor a fentiek alapján a reprezentációhoz tartozó rendezés szerinti j-edik csúcs akkor és csak akkor forrás, ha $\mathbf{r}_0$ j-edik komponense 0. Ennek alapján, $\mathbf{r}_0$ ismeretében nagyon egyszerűen ki tudunk választani egy forrást $\mathcal{G}$-ből. Másrészt vegyük észre, hogy amennyiben a j-edik csúcs a forrás, akkor ezen csúcs és a belőle kivezető élek törlése a reprezentációs mátrix j-edik sorának és j-edik oszlopának egyidejű törlésével ekvivalens, a törléssel előálló mátrix pontosan $\mathcal{G}_1$ reprezentációs mátrixa. Az új mátrixra vonatkozó oszlopösszegeket pedig úgy kapjuk meg, hogy a $\mathbf{r}_0$ vektorból kivonjuk a mátrix j-edik sorvektorát és a j-edik komponenst töröljük. Az előálló sorvektort jelölje $\mathbf{r}_1$. Ennek alapján a sorszámozás végrehajtásához tulajdonképpen a $\mathbf{r}_0, \ldots, \mathbf{r}_{n-1}$ vektorsorozatot kell előállítanunk, aminek a műveletigénye: $O(n^2)$.

Az eljárás illusztrálására tekintsük a következő feladatot.

2.2. példa. Legyen $\mathcal{G} = (V, E)$, ahol $V = \{a, b, c, d, e, f, g\}$, és az élek legyenek adottak a gráf alábbi mátrix-reprezentációjával.

$$\begin{pmatrix} & a & b & c & d & e & f & g \\ a & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ b & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ c & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ d & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ e & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ f & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ g & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

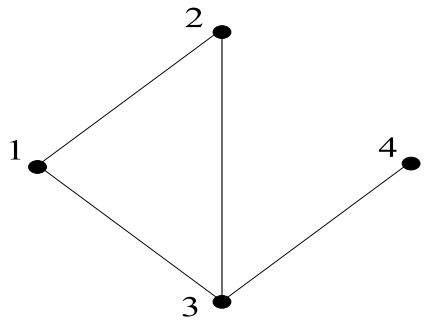
Akkor $\mathbf{r}_0 = (2, 2, 1, 3, 3, 0, 1)$, tehát az f csúcs forrás a $\mathcal{G}$ gráfban, így adjuk az f csúcsnak az 1 sorszámot. Véve f sorvektorát és kivonva $\mathbf{r}_0$-ból, $\mathbf{r}_1 = (2, 1, 0, 3, 3, -, 0)$ adódik, ahol a hatodik komponens törlését egy

" – " jellel jelöltük. Az $\mathbf{r}_1$ vektor a $\mathcal{G}_1$ gráf reprezentációs mátrixának oszlopösszegeit tartalmazza, így c és g források a $\mathcal{G}_1$ gráfban. Válasszuk közülük c-t, és adjuk a c csúcsnak a 2 sorszámot. Kivonva c sorvektorát $\mathbf{r}_1$-ből, $\mathbf{r}_2 = (1, 1, -, 3, 2, -, 0)$, azaz a g csúcs forrás $\mathcal{G}_2$-ben; adjuk neki a 3 sorszámot. Kivonva g sorvektorát $\mathbf{r}_2$-ből, $\mathbf{r}_3 = (1, 0, -, 2, 2, -, -)$, tehát a b csúcs forrás $\mathcal{G}_3$-ban, és b sorszáma 4 lesz. Folytatva az eljárást, $\mathbf{r}_4 = (0, -, -, 1, 2, -, -)$ így a lesz forrás $\mathcal{G}_4$-ben és az 5 sorszámot kapja, majd $\mathbf{r}_5 = (-, -, -, 0, 1, -, -)$, d a forrás $\mathcal{G}_5$-ben és d sorszáma 6, végül $\mathbf{r}_6 = (-, -, -, -, 0, -, -)$, ekkor e a forrás $\mathcal{G}_6$-ban, és e sorszáma 7 lesz.

Sok esetben a pontok és a köztük levő kapcsolatok nem irányítottak, hanem kétirányúak. Például, ha a tud telefonálni b-nek, akkor b is tud telefonálni a-nak. Az ehhez hasonló kapcsolatrendszerek leírására szolgálnak az irányítatlan gráfok.

Egy $\mathcal{G} = (V, E)$ párost *irányítatlan gráfnak* nevezünk, ahol V a gráf csúcspontjainak véges, nemüres halmaza, és $E \subseteq V \times V \setminus I_V$ a gráf éleinek a halmaza, továbbá minden $(u, v) \in E$ élre $(v, u) \notin E$ teljesül. Ha $(u, v) \in E$, akkor azt mondjuk, hogy u és v az (u, v) *él végpontjai*, továbbá az u és v *csúcsok illeszkednek az* (u, v) *élre*. Használni fogjuk az utóbbi esetben az (u, v) *él illeszkedik az* u *csúcsra és a* v *csúcsra* kifejezést is. Ha a $v \in V$ csúcs j számú élre illeszkedik, akkor azt mondjuk, hogy a v *csúcs fokszáma* j. Legyenek $v_1, \ldots, v_k$ $(k \geq 2)$ az irányítatlan $\mathcal{G}$ gráf páronként különböző csúcsai, továbbá legyen $e_t \in E$, $(t = 1, \ldots k - 1)$. Ha minden t-re e_t végpontjai v_t és v_{t+1}, akkor az $e_1, \ldots, e_{k-1}$ élsorozatot *irányítatlan útnak* nevezzük, melynek v_1 a *kezdőpontja* és v_k a *végpontja*. Ha van olyan, a fentiektől különböző $e \in E$ él, hogy a tekintett út kezdőpontja és végpontja e végpontjai, akkor a fenti élsorozatot ezzel az éllel kiegészítve, a kapott élsorozatot *irányítatlan körútnak* nevezzük, és azt mondjuk, hogy $\mathcal{G}$ tartalmaz irányítatlan kört. A $\mathcal{G}$ irányítatlan gráfot *körmentesnek* nevezzük, ha nem tartalmaz irányítatlan körutat. Azt mondjuk, hogy a $\mathcal{G}$ irányítatlan gráf *összefüggő*, ha bármely két csúcsa között vezet irányítatlan út. $\mathcal{G}$ egy maximális összefüggő részgráfját $\mathcal{G}$ egy *komponensének* nevezzük. A bevezetett fogalmakat az alábbi példán illusztráljuk.

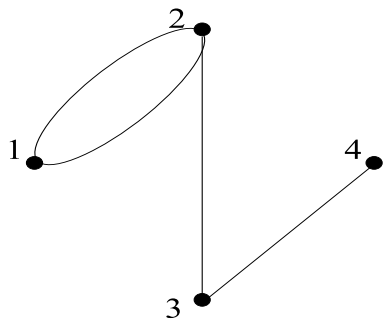
2.3. példa. Tekintsük a $\mathcal{G} = (\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (2, 3), (3, 4)\})$ irányítatlan gráfot. A gráf grafikus reprezentációját, melyben az élek irányítatlan szakaszok, mutatja a 2.2. ábra. A G gráf összefüggő, az $(1, 2)$ él, az $(1, 2), (1, 3)$ élsorozat irányítatlan utak, az $(1, 2), (2, 3), (1, 3)$ élsorozat pedig irányítatlan kör.



2.2. ábra. A 2.3. példa gráfjának grafikus reprezentációja.

Az irányítatlan gráfok általánosításának tekinthetők a multigráfok. Egy $\mathcal{G} = (V, E)$ párost *multigráfnak* nevezünk, ahol V a gráf csúcspontjainak véges, nemüres halmaza, E pedig $V \times V \setminus I_V$-beli élek véges rendszere, továbbá, ha $(u, v) \in E$, akkor $(v, u) \notin E$ teljesül E-re. Tehát a multigráf csak annyiban különbözik az irányítatlan gráftól, hogy bizonyos csúcspárokat egynél több éllel is össze lehet köti. A multigráfokat az alábbi példával szemléltetjük.

2.4. példa. Legyen $\mathcal{G} = (\{1, 2, 3, 4\}, \{(1, 2), (1, 2), (2, 3), (3, 4)\}$. Mivel az $(1, 2)$ él kétszer szerepel az E rendszerben, $\mathcal{G}$ multigráf, melynek grafikus reprezentációját mutatja a 2.3. ábra.



2.3. ábra. A 2.4. példa multigráfjának grafikus reprezentációja.

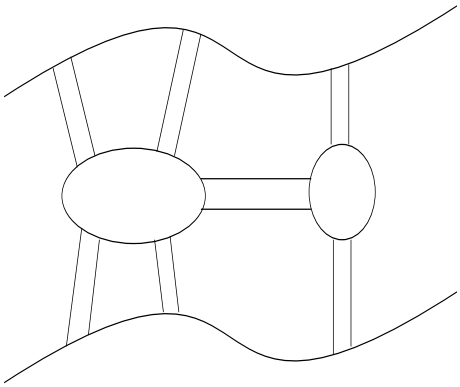
A multigráfokra ugyanúgy definiálhatók az olyan fogalmak, mint egy él végpontjai, csúcsok illeszkedése élre, él illeszkedése csúcsra, csúcs fokszáma, irányítatlan út, irányítatlan kör, összefüggőség, komponens, ahogy azt irányítatlan gráfokra megtettük. Ezért a továbbiakban ezeket a fogalmakat szabadon használjuk multigráfokra.

A multigráfok vizsgálatával kezdődött a gráfelmélet kialakulása. 1736-ban L. Euler [53] a königsbergi hidak problémáját tanulmányozta és oldotta meg multigráfokkal. A probléma a következő.

Königsbergi hidak problémája

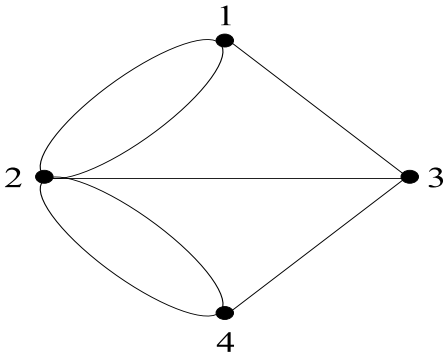
Hét híd köti össze a Pregel folyó két szigetét és partját. A kérdés az, hogy lehet-e valamelyik területről elindulva, oda visszatérni úgy, hogy a séta során minden hídon pontosan egyszer megyünk át.

Az alábbi ábra mutatja a szigetek és hidak elhelyezkedését.



2.4. ábra. A königsbergi hidak problémája.

Ha a folyónak az ábrán levő felső partját 1-gyel, az alsó partját 4-gyel, a baloldali szigetet 2-vel és a jobboldali szigetet 3-mal jelöljük, és a hidakat tekintjük az éleknek, akkor az alábbi multigráfhoz jutunk.



2.5. ábra. A königsbergi hidak problémájának multigráfja.

Ekkor a probléma a következőképpen fogalmazható meg. Vegyük a fenti multigráfban csúcsok egy olyan $v_1, \ldots, v_{m+1}$ sorozatát és a gráf éleinek egy olyan $e_{i_1}, \ldots, e_{i_m}$ permutációját, hogy a v_j, v_{j+1} csúcsok illeszkedjenek az e_{i_j} élhez, továbbá legyen $v_1 = v_{m+1}$. Az ilyen élsorozatot a gráf *Euler körének* nevezzük, és azt mondjuk, hogy a gráf *Euler-féle*, ha létezik Euler köre. Így a königsbergi hidak problémája visszavezethető annak eldöntésére, hogy a 2.5. ábrán megadott multigráf Euler-féle vagy sem. A kérdésre Euler [53] alábbi tétele adja meg a választ.

2.1. tétel. *Egy irányítatlan $\mathcal{G}$ gráf vagy multigráf akkor és csak akkor Euler-féle, ha $\mathcal{G}$ összefüggő és minden csúcsának páros a fokszáma.*

Bizonyítás. Elsőként tegyük fel, hogy $\mathcal{G}$ Euler-féle. Akkor bármely csúcspárra létezik őket összekötő irányítatlan út $\mathcal{G}$-ben, így $\mathcal{G}$ összefüggő. Másrészt bármely v csúcsra ahányszor az Euler kör befut a csúcsba, tovább is megy a csúcsból, így minden csúcs fokszáma páros, mert az Euler kör minden élet pontosan egyszer tartalmaz.

Most tegyük fel, hogy a tekintett irányítatlan gráf vagy multigráf összefüggő és minden csúcsának fokszáma páros. Megmutatjuk, hogy ekkor $\mathcal{G}$-nek létezik Euler köre. Ezt úgy tesszük, hogy megadunk egy eljárást, ami megkonstruál $\mathcal{G}$-ben egy Euler kört. Az eljárás a következő:

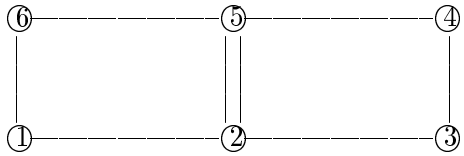
(1) Induljunk egy v csúcsból egy illeszkedő élen és haladjunk ameddig lehet még be nem járt éleken, közben sorszámozzuk a bejárt éleket. Csak v-ben akadhatunk el mivel minden csúcs fokszáma páros.

(2) Ha minden élet bejártunk, akkor vége az eljárásnak, a sorszámozott élsorozat Euler kör. Ha nem jártunk még be minden élet, akkor (3) következik.

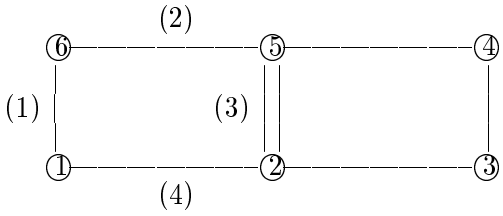
(3) Van olyan be nem járt él, amely valamely bejárt u ponthoz illeszkedik mivel $\mathcal{G}$ összefüggő. Folytassuk innen a bejárást a be nem járt éleken u újbóli eléréséig, és írjuk át a sorszámozást úgy, hogy v-től u-ig maradnak a sorszámok, aztán u-tól u-ig majd u-tól v-ig sorszámozunk. Ezután folytassuk az eljárást a (2) lépéssel.

A 2.1. tétel bizonyításában megadott eljárást a következő multigráfon demonstráljuk, ahol az élek sorszámait az élek felett illetve mellett tüntetjük fel.

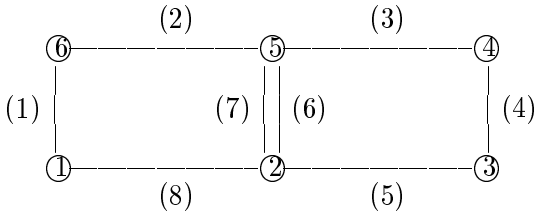
2.5. példa.



Első élsorozat: $(1,6),(5,6),(2,5),(1,2)$.



Második élsorozat: $u = 5,\ (4,5),(3,4),(2,3),(2,5)$.



Euler kör: $(1,6),(5,6),(4,5),(3,4),(2,3),(2,5),(2,5),(1,2)$.

Az egyszerűbb szóhasználat érdekében a továbbakban irányítatlan gráfokra és multigráfokra az út és a kör esetében eltekintünk az irányítatlan jelzőtől, út és kör alatt mindig irányítatlan utat és kört értünk.

A jelen fejezet lezárásaként megismerkedünk a hálózat fogalmával, annak különböző reprezentációival.

Hálózaton egy olyan irányított gráfot, irányítatlan gráfot vagy multigráfot értünk, amelynek minden éléhez hozzá van rendelve egy valós szám. Az élhez rendelt szám interpretálható különböző módon, attól függően, hogy a tekintett hálózattal kapcsolatban mi a célunk. Például, ha az éleket útszakaszoknak tekintjük, akkor az élhez rendelt szám megadhatja az illető útszakasz hosszát vagy időegységre vonatkozó átbocsájtó képességét. Minden esetben, amikor hálózatokról lesz szó, egyértelműen megadjuk az élekhez rendelt valós számok interpretációját. Ez azért fontos, mert bizonyos

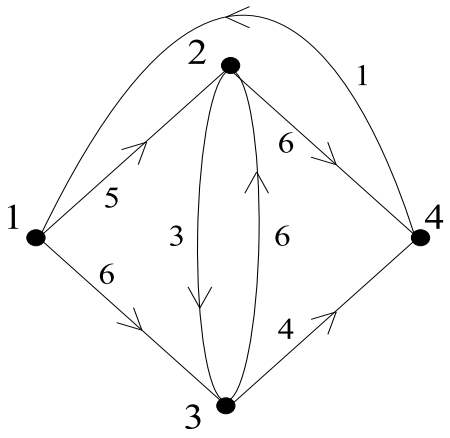
értelemben hatással van a hálózat mátrixreprezentációjára. A hálózatokat a gráfokhoz hasonlóan kétféle módon fogjuk reprezentálni.

A grafikus reprezentációban egyszerűen az élek fölé írjuk a hozzájuk rendelt valós számokat. Ezzel nyilvánvalóan minden, a hálózatra vonatkozó információ megadásra kerül.

Kicsit bonyolultabb a hálózat mátrix-reprezentációja. Kézenfekvőnek tűnik, a gráfok mátrixreprezentációjának egy olyan módosítása, hogy a reprezentációs mátrixba az 1 értékek helyett a megfelelő élhez tartozó valós számot írjuk. Ez kielégítő abban az esetben, ha egyik élhez sem rendeltünk 0 értéket. Ellenkező esetben viszont olyan gond adódik, hogy nem tudjuk megkülönböztetni a nem létező éleket azon élektől, amelyekhez 0 van rendelve, ami esetenként problémát okozhat. További gond, hogy a három eset (irányított gráf, irányítatlan gráf, multigráf) nem kezelhető egységesen. Ezek kiküszöbölésére a hálózat reprezentációs mátrixát a három esetre külön definiáljuk.

Elsőként tekintsük az irányított gráfokat. Tegyük fel, hogy a hálózat gráfja n számú csúcspontot tartalmaz és vegyük a gráf csúcsainak egy lineáris rendezését. Ezek után képezzük a következő (w_{ij}) $n \times n$-es mátrixot. Legyen $w_{ij} = c_{ij}$, ha az i-edik csúcspontból vezet él a j-edik csúcspontba, ahol a csúcsok sorszámozása a tekintett rendezés szerint történik, továbbá c_{ij} a tekintett élhez rendelt valós számot jelöli; ha nincs ilyen él, akkor legyen $w_{ij} = W$, ahol W egyelőre egy tetszőlegesen rögzített szimbólum, és ilyenkor *nemlétező élről* beszélünk. A (w_{ij}) mátrixot a tekintett *hálózat mátrix-reprezentációjának* nevezzük. Nyilvánvaló, hogy adott rendezés mellett a (w_{ij}) mátrix egyértelműen meghatározott. A W szimbólumot minden esetben a vizsgált problémától függően külön definiáljuk. Például, ha a valós számok úthosszakat jelölnek és minimális utakat akarunk meghatározni, akkor W-ét választhatjuk $+\infty$-nek. Ha az élekhez rendelt valós számok átbocsájtóképességet adnak meg, akkor W lehet 0, mivel a probléma szempontjából közömbös lesz, hogy egyáltalán nem létezik út két pont között vagy olyan út létezik, ami a 0 átbocsájtóképesség miatt nem használható. Illusztrációként tekintsük azt a hálózatot, amelynek a 2.6 ábra egy grafikus reprezentációja.

2.6. példa.



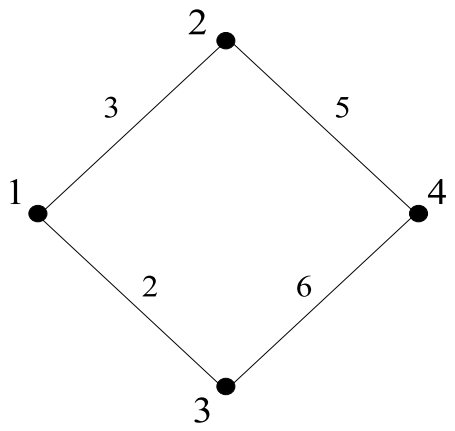
2.6. ábra. Egy irányított hálózat grafikus reprezentációja.

A grafikusan megadott irányított hálózat mátrix-reprezentációja a következő:

$$\begin{pmatrix} W & 5 & 6 & W \\ W & W & 3 & 6 \\ W & 6 & W & 4 \\ 1 & W & W & W \end{pmatrix}$$

Irányítatlan gráf esetén tegyük fel, hogy a csúcspontok halmaza $\{1, \ldots, n\}$. Mivel minden $i \neq j$ csúcspárra, az (i, j) vagy (j, i) párok közül legfeljebb csak az egyik lehet eleme az E élhalmaznak, ezért az általánosság megszorítása nélkül feltételezhetjük, hogy minden (i, j)-re $(i, j) \in E$-ből, $i < j$ következik. Ekkor a gráf mátrix-reprezentációja egy felső trianguláris bináris mátrix lesz. Most az éleket reprezentáló 1-esek helyébe beírhatjuk a illető élhez rendelt számot, míg a nemlétező élekre alkalmazhatjuk ugyanazt a W-és technikát, mint az irányított esetben. Ezt a reprezentációt az alábbi példával illusztráljuk.

2.7. példa. Tekintsük azt az irányítatlan hálózatot, amelynek grafikus reprezentációját adja meg a 2.7. ábra.



2.7. ábra. Egy irányítatlan hálózat grafikus reprezentációja.

A tekintett irányítatlan hálózat mátrix-reprezentációja a következő:

$$\begin{pmatrix} & 3 & 2 & W \\ & & W & 5 \\ & & & 6 \\ & & & \end{pmatrix}$$

Bár a multigráfos hálózatok mátrix-reprezentációját nem fogjuk használni, a teljesség kedvéért megadunk egy technikát az ilyen hálózatok mátrix-reprezentációjára. A reprezentáló mátrix egy felső triangularis mátrix lesz, amelynek az elemei vagy valós számok, vagy a W szimbólum vagy valós számokból álló vektorok lesznek. Konkrétan minden $i < j$ indexpárra,

(1) ha az (i, j) pár egy példányban szerepel a multigráfban, akkor legyen w_{ij} az illető élhez rendelt szám,

(2) ha az (i, j) pár egyetlen példánya sem szerepel a multigráfban, akkor legyen $w_{ij} = W$,

(3) ha az (i, j) pár k példányban szerepel a multigráfban és az egyes példányokhoz rendelt valós számok $r_1, \ldots, r_k$, akkor legyen w_{ij} egyenlő az $(r_1, \ldots, r_k)$ vektorral.

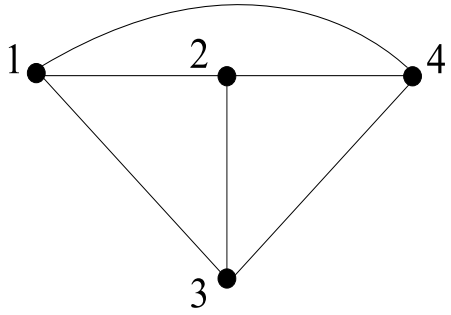
Legyenek adottak a $\mathcal{G}' = (V', E')$ és $\mathcal{G} = (V, E)$ irányítatlan gráfok (multigráfok). Tegyük fel, hogy az élek $\mathcal{G}'$-ben ugyanazon (u, v) párokkal vannak reprezentálva, mint $\mathcal{G}$-ben. Ezt az általánosság megszorítása nélkül

feltehetjük. A $\mathcal{G}'$ gráfot (multigráfot) $\mathcal{G}$ *részgráfjának* nevezzük, ha $V' \subseteq V$ és $E' \subseteq E$ (E' részrendszere E-nek) teljesül.

Egy $\mathcal{G}$ irányítatlan gráfot *fának* nevezünk, ha $\mathcal{G}$ összefüggő és körmentes. A fa definíciójából rögtön adódik, hogy bármely két csúcsa között pontosan egy út létezik a fában.

Egy $\mathcal{G} = (V, E)$ irányítatlan gráf vagy multigráf egy $\mathcal{G}' = (V', E')$ részgráfját *feszítőfának* nevezzük, ha $\mathcal{G}'$ fa és $V' = V$. Egyszerűen belátható, hogy $\mathcal{G}$-nek akkor és csak akkor van feszítőfája, ha $\mathcal{G}$ összefüggő. A feszítőfák illusztrálására tekintsük az alábbi példát.

2.8. példa. Legyen $\mathcal{G} = (\{1,2,3,4\}, \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\})$. A gráf grafikus reprezentációját mutatja a 2.8. ábra. Az $(1,4),(2,4),(3,4)$ és $(1,3),(1,2),(1,4)$ élekből álló fák mindegyike feszítőfája $\mathcal{G}$-nek.



2.8. ábra. A 2.8. példa $\mathcal{G}$ gráfjának grafikus reprezentációja.

Most egy olyan alkalmazást mutatunk be, amely kapcsolódik a feszítőfákhoz és az irányítatlan hálózatokhoz.

Csővezeték tervezése

Tételezzük fel, hogy valamely régióban adott n számú olajkút és egy finomítót építettek. Feladat egy minimális hosszúságú olyan csőhálózat kiépítése, amely összekapcsolja a kutakat a finomítóval.

A feladatot egy olyan irányítatlan hálózattal tudjuk modellezni, amelynek $n+1$ csúcspontja van, nevezetesen a kutak és a finomító. Minden csúcspár között vezet él, ez egy esetleges cső összeköttetés, továbbá az élekhez rendelt számok az éleknek megfelelő csőrendszerek hosszát adják meg. Ekkor a hálózat gráfjának egy minimális hoszúságú feszítőfája adja a feladat egy optimális megoldását, ahol a feszítőfa hosszán a benne szereplő élek hosszainak összegét értjük.

Az előző alkalmazás után természetesen vetődik fel a kérdés, hogy adott összefüggő gráfra (multigráfra), miként lehet egy minimális hosszúságú feszítőfát meghatározni. A megoldásra szolgáló eljáráshoz tekintsünk egy $\mathcal{G} = (V, E)$ összefüggő irányítatlan gráfot vagy multigráfot, amelyre $|V| = n$ és $|E| = m$. Továbbá tegyük fel, hogy $\mathcal{G}$ minden $e_t \in E$ éléhez hozzá van rendelve egy w_{e_t} valós szám, amit nevezzünk az illető él hosszának. $\mathcal{G}$ tetszőleges $\mathcal{G}'$ részgráfjára jelölje $w(\mathcal{G}')$ a $\mathcal{G}'$ részgráfba eső élekre az élhosszak összegét. Akkor az alábbi, J. B. Kruskaltól [115] származó eljárás szolgáltatja az optimális megoldást.

Kruskal eljárása [115]

- *1. lépés.* Legyen $T = \emptyset$, $j = 1$ és rendezzük az adott $\mathcal{G}$ éleit hosszaik szerint növekvő sorrendbe. Az eredmény legyen
$$w_{e_{i_1}} \leq \ldots \leq w_{e_{i_m}},$$
ahol w_e jelöli az $e \in E$ él hosszát.

- *2. lépés.* Ha $|T| = n - 1$, akkor vége az eljárásnak, $\mathcal{T} = (V, T)$ egy minimális hosszúságú feszítőfa.

  Ha $|T| < n - 1$ és $j \leq m$, akkor legyen $T' = T \cup \{e_{i_j}\}$. Ha T' nem tartalmaz kört, akkor legyen $T = T'$, $j = j + 1$ és ismételjük meg a 2. lépést.

  Ha T'-ben van kör, akkor $j := j + 1$ és ismételjük meg a 2. lépést.

Az eljárás helyességének igazolásához vegyük észre, hogy $m \geq n-1$ mivel $\mathcal{G}$ összefüggő. Állítjuk, hogy a 2. lépésben mindig lehet élet választani. Ezt indirekt igazoljuk. Tegyük fel, hogy valamely $|T| = k < n-1$-re nem tudunk élet választani, azaz az aktuális T élhalmazt nem lehet bővíteni. Jelölje U a T élhalmaz éleire illeszkedő csúcspontok halmazát. Akkor $|U| = k + 1 < n$. Mivel nem tudunk élet választani, ezért $\mathcal{G}$ minden olyan élére, amelynek valamely végpontja U-beli a másik végpontja is U-beli lesz. De akkor $\mathcal{G}$-ben nem vezet U-ból $V \setminus U$-ba él, ami ellentmond annak, hogy $\mathcal{G}$ összefüggő.

Jelölje a T-be rendre bevont élek sorozatát
$$e_{j_1}, \ldots, e_{j_{n-1}}.$$
Állítjuk, hogy a kapott $\mathcal{T}$ fa minimális hosszúságú. Az állítást indirekt igazoljuk. Ehhez tegyük fel, hogy $\mathcal{T}$ nem minimális hosszúságú. Ekkor van olyan $\mathcal{S}$ feszítőfa, hogy $w(\mathcal{S}) < w(\mathcal{T})$. Mivel $\mathcal{T} \neq \mathcal{S}$, van olyan legkisebb k, hogy $e_{j_k} = (u, v) \in T$ és $e_{j_k} \notin S$. Akkor $\mathcal{S}$-ben van pontosan egy olyan út,

amely u-ból v-be vezet. Ezért, ha az $e_{j_k} = (u, v)$ élet hozzávesszük $\mathcal{S}$-hez, akkor $\mathcal{S}$-ben egy $\mathcal{C}$ kört kapunk. Ha $\mathcal{C}$-nek minden éle benne lenne $\mathcal{T}$-ben, akkor ott is kört kapnánk, így $\mathcal{C}$-nek van olyan e_{i_l} éle , hogy $e_{i_l} \notin T$. $\mathcal{S}$-ben cseréljük ki az e_{i_l} élet e_{j_k}-ra, az új gráf legyen $\mathcal{S}'$. $\mathcal{S}'$ is feszítőfa és $w(\mathcal{S}) \leq w(\mathcal{S}')$, mivel $\mathcal{S}$ minimális hosszúságú. Ebből $w_{e_{i_l}} \leq w_{e_{j_k}}$ adódik. Most különböztessük meg az alábbi két esetet.

(1) $i_l < j_k$. Mivel $e_{i_l} \notin T$, e_{i_l} kört alkot az $e_{j_1}, \ldots, e_{j_{k-1}}$ élekkel. Másrészt k definíciójával ezek az élek mind S-ben vannak, de akkor $\mathcal{S}$ tartalmaz kört. Tehát ez az eset nem fordulhat elő, azaz $j_k < i_l$.

(2) $j_k < i_l$. Ebből az élek indexezése alapján

$$w_{e_{j_k}} \leq w_{e_{i_l}}$$

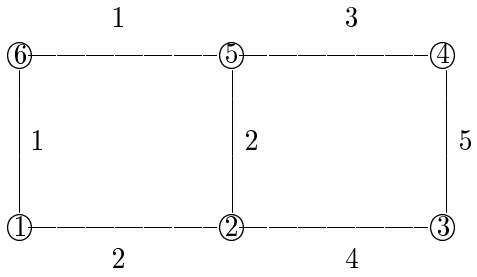
következik. De akkor $w_{e_{j_k}} = w_{e_{i_l}}$, amivel $w(\mathcal{S}) = w(\mathcal{S}')$ adódik, azaz $\mathcal{S}'$ is minimális hosszúságú feszítőfa.

Vegyük észre, hogy a $\mathcal{T}$ és az $\mathcal{S}'$ feszítő fák mindegyikében benne lesznek az $e_{j_1}, \ldots, e_{j_k}$ élek. Megismételve fentieket a $\mathcal{T}$ és $\mathcal{S}'$ feszítőfákkal, olyan $\mathcal{S}''$ minimális hosszúságú feszítőfát kapunk, hogy $\mathcal{T}$ és $\mathcal{S}''$ mindegyikében benne lesznek az $e_{j_1}, \ldots, e_{j_k}, e_{j_{k+1}}$ élek.

Folytatva az eljárást, véges sok lépés után egy olyan $\mathcal{S}^*$ feszítőfát kapunk, hogy $\mathcal{T} = \mathcal{S}^*$ és $w(\mathcal{S}^*) = w(\mathcal{S})$, ami ellentmond annak, hogy $w(\mathcal{S}) < w(\mathcal{T})$.

Az eljárást az alábbi irányítatlan hálózat minimális hosszúságú feszítőfájának meghatározásával mutatjuk be.

2.9. példa.



$w_{16} \leq w_{56} \leq w_{12} \leq w_{25} \leq w_{45} \leq w_{23} \leq w_{34}$

Az eljárás során előálló T halmazok:

$T_0 = \emptyset$,

$T_1 = \{(1, 6)\}$,

$T_2 = \{(1, 6), (5, 6)\}$,

$T_3 = \{(1,6),(5,6),(1,2)\}$,

a $(2,5)$ élet nem lehet bevonni,

$T_4 = \{(1,6),(5,6),(1,2),(4,5)\}$,

$T_5 = \{(1,6,),(5,6),(1,2),(4,5),(2,3)\}$.

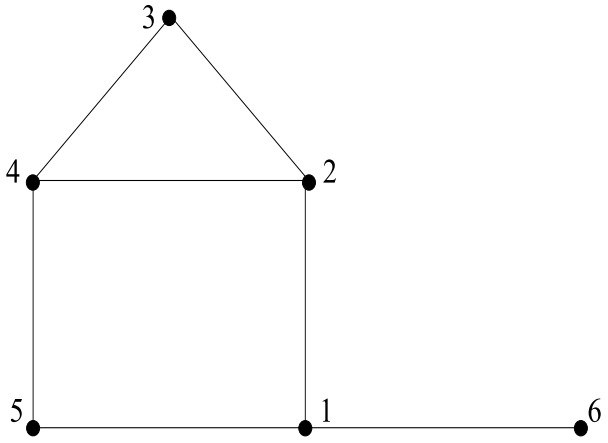
Ezzel befejeztük a gráfelméleti ismeretek felidézését. Azok számára, akik a gráfelmélet iránt érdeklődnek, ajánljuk a [3], [83], [178] munkákat.

3. Párosítási feladatok

A párosítási feladatok a gráfelméletből származó optimalizálási feladatok, melyeknek számos érdekes alkalmazása van. A jelen fejezetben, bizonyos előkészületek után, többféle párosítási feladattal is megismerkedünk. Az egyszerűbb szóhasználat érdekében ebben a fejezetben gráfon mindig irányítatlan gráfot értünk.

Legyen adott egy $\mathcal{G} = (V, E)$ gráf. $\mathcal{G}$ éleinek egy $M(\subseteq E)$ halmazát $\mathcal{G}$ egy *párosításának* nevezzük, ha $\mathcal{G}$ minden csúcspontja legfeljebb egy M-beli élnek végpontja.

3.1. példa. Legyen $\mathcal{G} = (V, E)$, ahol $V = \{1, \ldots, 6\}$ és az élek halmaza $E = \{(1,2), (1,5), (1,6), (2,3), (2,4), (3,4), (4,5)\}$. A gráf grafikus reprezentációját adja meg a 3.1. ábra. Akkor az $M = \emptyset$, $M' = \{(1,2), (4,5)\}$ és $M'' = \{(2,4)\}$ élhalmazok mindegyike párosítása $\mathcal{G}$-nek.



3.1. ábra. A 3.1. példa gráfjának grafikus reprezentációja.

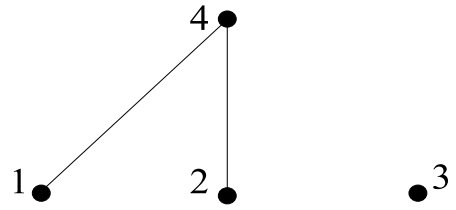
A $\mathcal{G} = (V, E)$ gráf egy $M \subseteq E$ párosítását *teljes párosításnak* nevezzük, ha a gráf minden csúcspontja végpontja valamelyik M-beli élnek.

Például a 3.1. példában szereplő $\mathcal{G}$ gráf $M = \{(1,6), (2,3), (4,5)\}$ élhalmaza $\mathcal{G}$ egy teljes párosítása.

Szükségünk lesz egy további új fogalomra.

Egy $\mathcal{G} = (V, E)$ irányítatlan gráfot *páros gráfnak* nevezünk, ha a csúcspontok V halmazának van olyan $\{A, B\}$ valódi osztályozása, hogy $\mathcal{G}$ minden élének a két végpontja különböző osztályban van. Szemléletesen, ha az A osztály elemeit kékre, a B osztály elemeit pirosra színezzük, akkor minden élnek az egyik végpontja kék, a másik végpontja pedig piros. Ha $\{A, B\}$ megfelelő osztályozás, akkor szokásos a $\mathcal{G}$ páros gráfot $\mathcal{G} = (A \cup B, E)$ formában megadni. Ha az A halmaz minden csúcsából vezet él a B halmaz minden csúcsába, azaz minden egyes kékre színezett csúcsból vezet él minden piros csúcsba, akkor azt mondjuk, hogy $\mathcal{G}$ *teljes páros gráf.*

3.2. példa. Tekintsük a $\mathcal{G} = (\{1, \ldots, 4\}, E)$ gráfot, ahol az élek halmaza $E = \{(1, 4), (2, 4)\}$. A gráf grafikus reprezentációját adja meg a 3.2. ábra. Akkor egyszerűen ellenőrizhető, hogy $\mathcal{G}$ páros gráf, két alkalmas osztályozása van $\mathcal{G}$ csúcspontjainak, mégpedig az $\{\{1, 2\}, \{3, 4\}\}$ és $\{\{1, 2, 3\}, \{4\}\}$ osztályozások. Ha kiegészítjük a tekintett gráfot a $(3, 4)$ éllel, akkor egy teljes páros gráfot kapunk, ahol a megfelelő osztályozás $\{\{1, 2, 3\}, \{4\}\}$.



3.2. ábra. A 3.2. példa gráfjának grafikus reprezentációja.

Most már készek vagyunk párosítási feladatok vizsgálatára.

Elsőként maximális számú élből álló párosításokat keresünk gráfokban. A következő alfejezetben olyan páros gráfokban vizsgálunk teljes párosításokat, amelyekben a gráf párosságát biztosító osztályok elemszáma azonos. Továbbá a gráf minden élének lesz egy súlya, azaz hálózatokat tekintünk, és a teljes párosítások között keresünk minimális súlyút, ahol az M *párosítás súlya* alatt az M-bei élek súlyainak összegét értjük.

Ezt követően, a harmadik alfejezetben tetszőleges gráfban vizsgáljuk a teljes párosításokat, és ezek között keresünk minimális súlyút. Ez technikailag igen bonyolult eljárással oldható meg, ezért ezt az eljárást csak nagy vonalakban érintjük.

A párosítási feladatok iránt érdeklődők számára ajánljuk a [112] és [129] könyveket.

3.1. Maximális élszámú párosítási probléma

Legyen adott egy $\mathcal{G} = (V, E)$ gráf. Határozzuk meg $\mathcal{G}$ egy maximális élszámú párosítását.

A fenti feladatban $\mathcal{G}$ párosításai a lehetséges megoldások és a z célfüggvény minden párosításhoz hozzárendeli annak élszámát. Ezért a feladatot felírhatjuk a következő formában is:

(3.1.1.) $$\max\{|M| : M \ \mathcal{G} \text{ párosítása } \}$$

Mivel $M = \emptyset$ párosítás bármely $\mathcal{G}$ gráfban és adott $\mathcal{G}$ gráf párosításainak száma véges, ezért a (3.1.1) feladatnak mindig létezik optimális megoldása.

Nyilvánvaló, hogy a 3.1. példában megadott $\mathcal{G}$ esetén bármely 3 élből álló párosítás maximális élszámú párosítás. Megjegyezzük, hogy általában egy feladatnak több maximális élszámú párosítása is lehet.

A maximális élszámú párosítási feladatnak számos alkalmazása van. Ezek közül most bemutatunk egyet.

Maximális számú játszma problémája

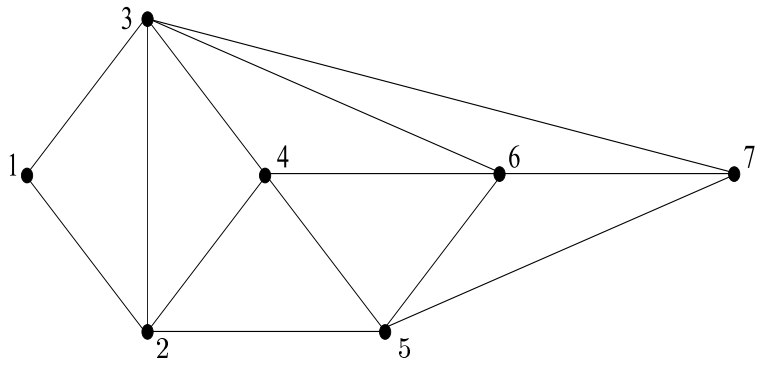
Adott egy társaság, melynek tagjai sakkozni kívánnak egymással. Ismert, hogy ki kivel hajlandó sakkozni. Feladat a társaság tagjait úgy összepárosítani, hogy a párok tagjai kölcsönösen hajlandók legyenek játszani egymással és a játszmák száma maximális legyen.

A probléma a következő párosítási feladatra vezethető vissza. Tekintsük azt a gráfot, amelynek csúcsai a társaság tagjai. Két személy akkor és csak akkor legyen éllel összekötve, ha kölcsönösen hajlandók játszani egymással. Akkor nyilvánvaló, hogy egy maximális élszámú párosítás a fenti feladatnak egy optimális megoldása.

A továbbiakban szükségünk lesz egy új fogalomra. Ennek bevezetéséhez legyen adott egy $\mathcal{G} = (V, E)$ gráf és legyen $M(\subseteq E)$ a $\mathcal{G}$ gráf egy párosítása. A $\mathcal{G}$ gráf egy v csúcsát *M-re nézve szabad csúcsnak* nevezzük, ha v nem végpontja M egyetlen élének sem. A $\mathcal{G}$ gráf egy $(v_1, v_2), \ldots, (v_{k-1}, v_k)$ útját *M-re nézve alternáló útnak* nevezünk, ha az útban szereplő élek felváltva szerepelnek az M és $E \setminus M$ halmazokból. Egy $(v_1, v_2), \ldots, (v_{k-1}, v_k)$ M-re nézve alternáló utat *M-re nézve javító útnak* nevezünk, ha v_1 és v_k nem végpontja egyetlen M-beli élnek sem.

Az alternáló és javító utak szemléltetésére tekintsük a következő példát.

3.1.1. példa. Legyen $\mathcal{G} = (\{1, \ldots, 7\}, E)$, ahol $E = \{(1,2), (1,3), (2,3), (2,4), (2,5), (3,4), (3,6), (3,7), (4,5), (4,6), (5,6), (5,7), (6,7)\}$. A gráf grafikus reprezentációját mutatja a 3.1.1. ábra. Továbbá tekintsük a $\mathcal{G}$ gráf $M = \{(2,4), (3,6)\}$ párosítását. Akkor például a $\mathcal{P}_1 = (2,4), (4,5)$, $\mathcal{P}_2 = (5,7)$, $\mathcal{P}_3 = (1,2), (2,4), (4,5)$ utak mindegyike M-re nézve alternáló út, és $\mathcal{P}_2$, $\mathcal{P}_3$ M-re nézve javító utak.



3.1.1. ábra. A 3.1.1. példa gráfjának grafikus reprezentációja.

A javító út elnevezést a következő indokolja. Legyen adott egy M párosítás és egy $\mathcal{P}$ M-re nézve javító út. Hagyjuk el M-ből a $\mathcal{P}$-beli éleket, jelölje a kapott élhalmazt $\bar{M}$. Az $\bar{M}$ élhalmazhoz hozzávéve a $\mathcal{P}$ útból a nem M-beli éleket, az így előálló M' élhalmaz $\mathcal{G}$-nek egy olyan M' párosítása, melyre $|M'| = |M|+1$. Az M' párosítást *M $\mathcal{P}$-vel történő javításának* nevezük.

Szemléltetésként tekintsük a 3.1.1. példa $\mathcal{G}$ gráfjának M párosítását. M-re nézve javító út a $\mathcal{P}_3 = (1,2), (2,4), (4,5)$ út. Akkor M $\mathcal{P}_3$-mal történő M' javítását úgy kapjuk meg, hogy elhagyjuk M-ből a $(2,4)$ élet és a megmaradó élhalmazhoz, ami most $\{(3,6)\}$, hozzávesszük az $(1,2)$ és $(4,5)$ éleket. Tehát $M' = \{(1,2), (4,5), (3,6)\}$.

A javító út fogalmával a (3.1.1.) feladat megoldására kezdeményezhető egy olyan iterációs eljárás, amelyben kiindulunk egy M párosításból és keresünk M-re nézve javító utat. Ha találunk ilyen javító utat, akkor ezzel megváltoztatjuk M-et, majd az új párosítást tekintve M-nek rátérünk a következő iterációra. Egy ilyen eljárás felépítéséhez meg kell válaszolni a következő kérdéseket:

(1) miként lehet egy induló M párosítást meghatározni?

(2) mit állíthatunk M-ről, ha nem létezik M-re nézve javító út?

(3) hogyan lehet egy adott M párosításhoz javító utat meghatározni?

Az (1) kérdés egyszerűen megoldható. Lehet $M = \emptyset$ az induló párosítás vagy valamilyen egyszerű eljárással meghatározhatjuk $\mathcal{G}$ egy párosítását. (Például adott csúcsból kiindulva, utat képezünk $\mathcal{G}$-ben, és felvesszük M-be ezen út első, harmadik, ... élét.)

A második kérdésre ad választ a következő tétel, amelyet C. Berge [21] publikált 1957-ben.

3.1.1. tétel ([21]). *$\mathcal{G}$ egy M párosítása akkor és csak akkor maximális élszámú párosítás, ha nem létezik M-re nézve javító út $\mathcal{G}$-ben.*

Bizonyítás. Elsőként tegyük fel, hogy M $\mathcal{G}$ egy maximális élszámú párosítása. Ekkor nem létezhet javító út M-re nézve, ugyanis a javító úttal megváltoztatva M-et, $\mathcal{G}$-nek egy nagyobb élszámú párosítását kapnánk, mint $|M|$, ami ellentmondás.

Most tegyük fel, hogy $\mathcal{G}$-ben nem létezik javító út M-re nézve. Állítjuk, hogy M $\mathcal{G}$ egy maximális élszámú párosítása. Az állítást indirekt igazoljuk. Ha M nem maximális élszámú párosítás, akkor $\mathcal{G}$-nek vannak olyan maximális élszámú párosításai, melyek élszáma nagyobb, mint $|M|$. Legyen ezek közül M^* $\mathcal{G}$ egy olyan párosítása, amelynek a legtöbb közös éle van M-mel. Ekkor nyilvánvalóan $M \neq M^*$. Tekintsük azon élek E' halmazát, amelyek az M és M^* halmazok valamelyikében, de csak egyikében vannak. $E' \neq \emptyset$, ugyanis, ha $E' = \emptyset$, akkor az $M = M^*$ ellentmondáshoz jutnánk. Jelölje V' az E'-beli élek végpontjainak a halmazát és legyen $\mathcal{G}' = (V', E')$. Vegyük észre, hogy $\mathcal{G}'$ minden csúcsa legfeljebb két E'-beli élnek végpontja. Valóban, ellenkező esetben az illető csúcs az M és M^* párosítások valamelyikében két él végpontja lenne, ami ellentmondás. Ebből az következik, hogy $\mathcal{G}'$ minden komponense vagy út vagy egy kör. Most megmutatjuk, hogy $\mathcal{G}'$ egyetlen komponense sem lehet kör. Ezt indirekt igazoljuk.

Elsőként tegyük fel, hogy $\mathcal{G}'$ valamely komponense páratlan számú élből álló kör. Ekkor ezen kör valamely csúcsa végpontja lenne két élnek vagy M-ben vagy M^*-ban, ami ellentmondás. Következésképpen $\mathcal{G}'$ egyetlen komponense sem lehet páratlan számú élből álló kör.

Most tegyük fel, hogy $\mathcal{G}'$ valamely komponense páros számú élből álló kör. Mivel mind M, mind M^* $\mathcal{G}$ párosításai, a tekintett körben alternálva fordulnak elő az élek. Most vegyük ki M^*-ból a körbeli éleit és a kapott élhalmazhoz vegyük hozzá azokat az éleket, amelyek a körnek is és M-nek is elemei. A kapott élhalmazt jelölje M'. Akkor M' $\mathcal{G}$ egy párosítása, továbbá $|M^*| = |M'|$, azaz M' $\mathcal{G}$ egy maximális élszámú párosítása. Vegyük észre, hogy M'-nek legalább eggyel több közös éle van M-mel, mint M^*-nak, ami

ellentmond M^* definíciójának. Tehát $\mathcal{G}'$ egyetlen komponense sem lehet páros számú élből álló kör.

A fentiek alapján $\mathcal{G}'$ minden komponense út. Tekintsünk most egy ilyen utat. Ha az út első és utolsó éle M-beli, akkor ez az út javító út M^*-ra nézve, ami ellentmond annak, hogy M^* maximális élszámú párosítás. Ha az út első és utolsó éle M^*-beli, akkor az út javító út M-re nézve, ami ellentmond annak a feltevésünknek, hogy nem létezik M-re nézve javító út. Végül, ha a tekintett élek egyike M-beli a másik pedig M^*-beli, akkor hasonlóan a páros körök esetéhez konstruálhatunk az út alapján M^*-ból $\mathcal{G}$-nek egy olyan M' maximális élszámú párosítását, amely legalább eggyel több közös élet tartalmaz M-mel, mint M^*-gal, ami ismét ellentmond M^* definíciójának. Ezzel a 3.1.1. tétel bizonyítását befejeztük.

Ezek után már csak a (3) kérdésre kell megadni a választ. Adott M párosításhoz javító út keresésére J. Edmonds [47] adta meg az első eljárást 1965-ben. Ezzel fogunk megismerkedni a következőkben. Az eljáráshoz szükségesek bizonyos előkészületek.

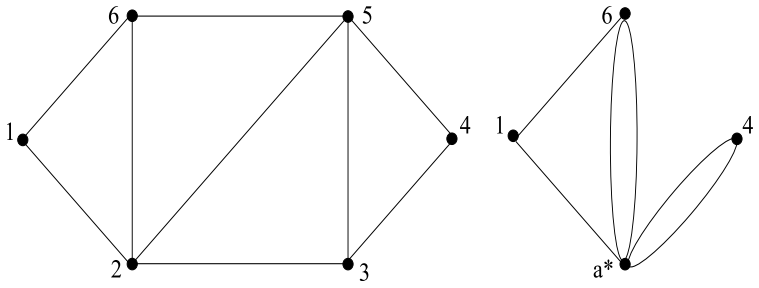
Elsőként vegyük észre, hogy a javító út definíciójából következik, hogy annak kezdő csúcsa és befejező csúcsa M-re nézve szabad. Ez az alábbi észrevételeket eredményezi.

(i) Ha nincs M-re nézve szabad csúcs $\mathcal{G}$-ben, akkor M maximális élszámú párosítása $\mathcal{G}$-nek. (Ilyenkor $\mathcal{G}$ minden csúcsa illeszkedik valamelyik M-beli élre, azaz az M párosítás $\mathcal{G}$ egy teljes párosítása lesz. Nyilvánvaló, hogy ez az eset csak akkor fordulhat elő, ha $|V|$ páros.)

(ii) Ha $\mathcal{G}$-ben csak egy M-re nézve szabad csúcs van, akkor M maximális élszámú párosítása $\mathcal{G}$-nek. Ha nem így lenne, akkor a 3.1.1. tétel alapján létezne M-re nézve javító út $\mathcal{G}$-ben. De akkor a javító út két végpontja M-re nézve szabad csúcs lenne, ami ellentmondás.

Legyen a $\mathcal{G} = (V, E)$ gráf vagy multigráf egy páratlan számú élet tartalmazó köre $\mathcal{C}$. A $\mathcal{G}$ *$\mathcal{C}$-vel való zsugorításának* nevezzük azt a $\mathcal{G}' = (V', E')$ gráfot vagy multigráfot, melynek csúcspontjait úgy kapjuk meg, hogy az eredeti csúcspontokból elhagyjuk a C körben lévő csúcsokat és a kapott A halmazhoz hozzáveszünk egy a^* mesterséges csúcspontot, azaz $V' = A \cup \{a^*\}$. Az E' élhalmaz egyrészt azon E-beli élekből áll, amelyeknek mindkét végpontja eleme A-nak, másrészt minden $(u, v) \in E$ élre felvesszük még az (u, a^*) élet E'-be, amennyiben $u \in A$ és v valamely csúcsa C-nek. (Ha a zsugorítással előálló $\mathcal{G}'$ multigráf, akkor egy adott él különböző példányait különböző élekként kezeljük, és az egyes példányokhoz hozzárendeljük indexként azon él végpontjait, amelyből az illető példány keletkezett.) A következő példában egy körzsugorítást mutatunk be.

3.1.2. példa. Legyen $\mathcal{G} = (\{1, \ldots, 6\}, E)$, ahol $E = \{(1,2), (1,6), (2,3), (2,5), (2,6), (3,4), (3,5), (4,5), (5,6)\}$. Vegyük $\mathcal{G}$-ben a $(2,3), (3,5), (2,5)$ élekből álló C kört. Akkor a $\mathcal{G}$ C-vel való $\mathcal{G}'$ zsugorítására $V' = \{1, 4, 6, a^*\}$ és $E' = \{(1,6), (1,a^*)_{1,2}, (6,a^*)_{2,6}, (6,a^*)_{5,6}, (4,a^*)_{4,5}, (4,a^*)_{3,4}\}$. A $\mathcal{G}$ gráfot és C-vel való zsugorítását mutatja a 3.1.2. ábra.



3.1.2. ábra. A 3.1.2. példa $\mathcal{G}$ gráfja és annak $\mathcal{G}'$ zsugorítása.

Nyilvánvaló hogy az indexezés felhasználásával $\mathcal{G}'$ből egyértelműen rekonstruálható a $\mathcal{G}$ gráf. Ilyenkor azt mondjuk, hogy $\mathcal{G}$ a $\mathcal{G}'$ *a^*-ra vonatkozó rekonstrukciója.*

Legyen adott egy $\mathcal{G}$ gráf (multigráf) és annak egy M párosítása, valamint legyen $\mathcal{C}$ $\mathcal{G}$ egy olyan $2k+1$ élszámú köre, amelyben k számú M-beli él van. Továbbá legyen $\mathcal{G}'$ $\mathcal{G}$-nek $\mathcal{C}$-vel való zsugorítása. Vegyük észre, hogy M meghatároz $\mathcal{G}'$-ben egy M' párosítást a következő módon. Elsőként vegyük fel M'-be M összes olyan élét, melyeknek egyik végpontja sem eleme a $\mathcal{C}$ körnek. $\mathcal{C}$ struktúrája miatt legfeljebb csak egy olyan M-beli (u,v) él lehet, amelynek egyik végpontja $\mathcal{C}$-beli (legyen ez a végpont v), és a másik végpontja (u) nem eleme a $\mathcal{C}$ körnek. Ha létezik ilyen él, akkor vegyük fel még M'-be az $(u,a^*)_{u,v}$ élet. Nyilvánvaló, hogy M' egy párosítása lesz $\mathcal{G}'$-nek. Az így képezett M' párosítást az *M által indukált párosításnak* nevezzük. Például, ha a 3.1.2. példában szereplő $\mathcal{G}$ gráfnak tekintjük az $M = \{(1,6), (2,5), (3,4)\}$ párosítását, akkor az M által indukált párosítás $\mathcal{G}'$-ben $M' = \{(1,6), (4,a^*)_{3,4}\}$. A megfordítás is igaz. Nevezetesen, ha $\mathcal{G}'$ $\mathcal{G}$-nek egy $2k+1$ élszámú $\mathcal{C}$ körrel való zsugorítása, és M' a $\mathcal{G}'$ gráf egy párosítása, akkor M'-t felhasználva, képezhető $\mathcal{G}$-nek egy $|M'|+k$ élet tartalmazó M párosítása a következő módon.

Elsőként vegyük fel M-be mindazon M'-beli éleket, amelyek nem illeszkednek a^*-ra.

Mivel M' párosítása $\mathcal{G}'$-nek, ezért legfeljebb egy olyan M'-beli él lehet, amelynek valamelyik végpontja a^*.

Ha nincs ilyen él, akkor egészítsük ki M-et a $\mathcal{C}$ körben lévő k számú olyan

éllel, amelyek páronként nem illeszkednek egymásra, például tekinthetjük a kör valamely w pontját és vehetjük az órajárással megegyező sorrendben w-ből indulva, a $\mathcal{C}$ kör első, a harmadik, $\ldots, 2k-1$-edik éleit.

Ha van ilyen él (legyen ez $(u, a^*)_{u,v}$), akkor vegyük fel M-be az (u, v) élet. Legyen az órajárással megegyező sorrendben a $\mathcal{C}$ kör v-re illeszkedő élének másik végpontja w. Most egészítsük ki az M halmazt w-ből indulva, az órajárással megegyező sorrendben haladva a $\mathcal{C}$ körön, az első, a harmadik, $\ldots, 2k-1$-edik élekkel.

Egyszerűen belátható, hogy M $\mathcal{G}$-nek egy $|M'|+k$ számú élet tartalmazó párosítása. Az így képezett M párosítást *az M' által generált párosításnak* nevezzük. Például, ha a 3.1.2. példában szereplő $\mathcal{G}'$ gráfban tekintjük az $M' = \{(1,6), (4,a^*)_{4,5}\}$ párosítást, akkor az általa generált párosítás $M = \{(1,6), (4,5), (2,3)\}$.

Szükségünk lesz a következő eljárásra.

Az eljáráshoz legyen adott egy $\mathcal{G}$ gráf vagy multigráf és annak egy M párosítása. Az eljárás során egy M-re nézve szabad s csúcsból kiindulva, egy alternáló fát építünk fel. A faépítés során a gráf éleit valamint a fa csúcsait megcímkézzük.

Alternáló fa eljárás ([47])

Előkészítő rész

Válasszunk egy M-re nézve szabad $s \in V$ csúcspontot. Legyen s a $\mathcal{T}_0$ fa gyökere és lássuk el a fában s-t az *outer* címkével. (Az összes többi csúcs és E minden éle egyelőre címkézetlen.) Legyen $r = 0$ és folytassuk az eljárást az iterációs eljárásrésszel.

Iterációs rész, faépítés (r-edik iteráció)

- *1. lépés.* Válasszunk E-ből egy olyan (u, v) címkézetlen élet, amely illeszkedik a $\mathcal{T}_r$ fa valamely u *outer* címkéjű csúcsára.

  Ha nincs ilyen él, akkor folytassuk az eljárást a 3. lépéssel.

  Ha (u, v) ilyen él, akkor v-től függően különböztessük meg a következő három esetet.

  (a) A v csúcs címkézetlen. Ekkor bővítsük a $\mathcal{T}_r$ fát az (u, v) éllel és E-ben címkézzük (u, v)-t az *igen* címkével.
  Ha v M-re nézve szabad csúcs, akkor vége az eljárásnak, $\mathcal{T}_r$-ben az s-ből a v-be vezető út egy javító út M-re nézve.

Ha v illeszkedik egy $(v,t) \in M$ élre, akkor bővítsük az aktuális $\mathcal{T}_r$ fát a (v,t) éllel, és az új fában lássuk el v-t az *inner* és t-t az *outer* címkékkel, továbbá lássuk el a (v,t) élet E-ben az *igen* címkével. Legyen $\mathcal{T}_{r+1} = \mathcal{T}_r$ és növeljük r értékét 1-gyel, majd térjünk rá a következő iterációs lépésre.

(b) A v csúcs címkézett és a címkéje *inner*. Ekkor lássuk el az (u,v) élet a *nem* címkével, és ismételjük meg az 1. lépést.

(c) A v csúcs címkézett és a címkéje *outer*. Ekkor lássuk el E-ben az (u,v) élet az *igen* címkével és folytassuk az eljárást a 2. lépéssel.

- *2. lépés.* (Páratlan élszámú kör esete.) Ebben az esetben (u,v) hozzávétele a $\mathcal{T}_r$ fához egy páratlan számú élből álló kör kialakulását eredményezi, ugyanis az u-ból a v-be vezető úton az *outer* és *inner* címkék felváltva követik egymást. Az eljárás itt befejeződik azzal, hogy felfedezett egy páratlan kört.

- *3. lépés.* (Magyar fa.) Ebben az esetben olyan fát kapunk, amelynek minden levele az *outer* címkével van ellátva, azaz a levelekre illeszkedő élek mindegyike M-beli él. (Ilyenkor nem létezik s-ből induló M-re nézve javító út.)

Nem igazoljuk, de belátható, hogy az eljárás véges lépésben véget ér és három eredményt adhat, nevezetesen M-re nézve javító utat, páratlan élszámú kört, és magyar fát.

Megjegyezzük, hogy az (u,v) él 1. lépésben történő vizsgálatakor (u,v) hozzávétele a $\mathcal{T}_r$ fához eredményezhetné egy páros számú élből álló kör kialakulását, de ezt az 1. lépésben az (u,v) *nem* címkével történő ellátásával kizárjuk. (Belátható, hogy ebben az esetben az (u,v) él nem fordulhat elő egyetlen s-ből induló javító útban sem.)

Az alternáló fa eljárás végrehajtását a 3.1.2. példa $\mathcal{G}$ gráfjának felhasználásával több párosításon szemléltetjük.

(1) Legyen $M = \{(2,3),(5,6)\}$. Akkor az 1 csúcs szabad M-re nézve.

$\mathcal{T}_0 = \{1\}$ és 1 *outer* címkét kap $\mathcal{T}_0$-ban.

1. lépés. Kiválasztjuk az $(1,6)$ élet. Az $(1,6),(5,6)$ élekkel bővítjük a $\mathcal{T}_0$ fát, és mindkét él *igen* címkét kap $\mathcal{G}$-ben, továbbá a 6 csúcs *inner*, az 5 csúcs pedig *outer* címkét kap $\mathcal{T}_0$-ban. $\mathcal{T}_1 = \{(1,6),(5,6)\}$. Növeljük r értékét 1-gyel és rátérünk a következő iterációs lépésre.

1. lépés. Kiválasztjuk az $(1,2)$ élet. Az $(1,2),(2,3)$ élekkel bővítjük a $\mathcal{T}_1$ fát, és mindkét élet ellátjuk *igen* címkével $\mathcal{G}$-ben, továbbá a 2 csúcs *inner*, a

3 csúcs *outer* címkét kap $\mathcal{T}_1$-ben. $\mathcal{T}_2 = \{(1,6),(5,6),(1,2),(2,3)\}$. Növeljük r értékét 1-gyel és rátérünk a következő iterációs lépésre.

1. lépés. Kiválasztjuk a $(4,5)$ élet. A 4 csúcs szabad csúcs M-re nézve, így vége az eljárásnak, az $(1,6),(5,6),(4,5)$ út javító út M-re nézve.

(2) Legyen ismét $M = \{(2,3),(5,6)\}$. Akkor az 1 csúcs szabad M-re nézve.

$\mathcal{T}_0 = \{1\}$ és 1 *outer* címkét kap $\mathcal{T}_0$-ban.

1. lépés. Kiválasztjuk az $(1,6)$ élet. Az $(1,6),(5,6)$ élekkel bővítjük a $\mathcal{T}_0$ fát, és mindkét él *igen* címkét kap $\mathcal{G}$-ben, továbbá a 6 csúcs *inner*, az 5 csúcs pedig *outer* címkét kap $\mathcal{T}_0$-ban. $\mathcal{T}_1 = \{(1,6),(5,6)\}$. Növeljük r értékét 1-gyel és rátérünk a következő iterációs lépésre.

1. lépés. Kiválasztjuk a $(3,5)$ élet. A $(3,5),(2,3)$ élekkel bővítjük a $\mathcal{T}_1$ fát, és mindkét élet ellátjuk *igen* címkével $\mathcal{G}$-ben, továbbá a 3 csúcs *inner*, a 2 csúcs *outer* címkét kap $\mathcal{T}_1$-ben. $\mathcal{T}_2 = \{(1,6),(5,6),((3,5),(2,3)\}$. Növeljük r értékét 1-gyel és rátérünk a következő iterációs lépésre.

1. lépés. Kiválasztjuk a $(2,5)$ élet. Most az 5 csúcs címkézett és a címkéje *outer*, így a (c) esetet kapjuk, a 2. lépéssel folytatjuk az algoritmust, ami kijelzi a $(2,3),(2,5),(3,5)$ élekből álló kört és vége az eljárásnak.

J. Edmonds a $\mathcal{G} = (V,E)$ gráf maximális élszámú párosításának meghatározására egy olyan eljárást dolgozott ki, amelyben kiindulva az $M = \emptyset$ párosításból, minden iterációs lépésben az alternáló fa eljárás egy vagy többszöri alkalmazásával meghatároz egy javító utat, amellyel javítja az aktuális M párosítást vagy kiderül, hogy az aktuális M-re nézve nem létezik javító út, azaz a 3.1.1. tétel alapján ekkor M egy maximális élszámú párosítás.

Edmonds eljárása ([47])

Előkészítő rész
Legyen $M_0 = \emptyset$ és $r = 0$.

Iterációs rész (r. iteráció)

- *1. lépés.* (Ellenőrzés.) Ha $\mathcal{G}$-ben egy csúcs kivételével minden M_r-re nézve szabad csúcs átvizsgált, akkor vége az eljárásnak, M_r maximális élszámú párosítás $\mathcal{G}$-ben. Ellenkező esetben legyen $i = 0$ és folytassuk az eljárást a 2. lépéssel. (Az i változóval számoljuk az iterációs lépesen belüli zsugorításokat.)

- *2. lépés.* Válasszunk $\mathcal{G}$-ben egy v szabad csúcsot M_r-re nézve. Legyen $\mathcal{G}_i = \mathcal{G}$, $\bar{M}_i = M_r$, $s = v$ és térjünk rá a 3. lépésre.

- *3. lépés.* Adjuk az s csúcsnak az *outer* címkét és indítsuk az alternáló fa eljárást $\mathcal{G}_i$-re és $\bar{M}_i$-re az s gyökérből kiindulva.

  Ha az alternáló fa eljárás $\bar{M}_i$-re nézve egy $\mathcal{P}_i$ javító utat eredményez $\mathcal{G}_i$-ben, akkor térjünk rá a 4. lépésre.

  Ha az alternáló fa eljárás magyar fát eredményez, akkor az 5. lépés következik.

  Ha az alternáló fa eljárás páratlan élszámú kör megtalálásával végződik, akkor a 6. lépés következik.

- *4. lépés.* (Javítás.) Javítsuk az $\bar{M}_i$ párosítást a megtalált $\mathcal{P}_i$ javító úttal, jelölje a javítás utáni új párosítást R_i. Határozzuk meg az R_i által generált párosítást $\mathcal{G}_{i-1}$-ben, amit jelöljön R_{i-1}. Majd határozzuk meg az R_{i-1} által generált párosítást $\mathcal{G}_{i-2}$-ben, amit jelöljön R_{i-2}, és folytassuk ezt addig, amíg megkapunk egy R_0 párosítást $\mathcal{G}_0$-ban. Legyen $M_{r+1} = R_0$. Növeljük r értékét 1-gyel, és folytassuk az eljárást a következő iterációs lépéssel.

- *5. lépés.* (Magyar fa.) Minősítsük a v csúcsot átvizsgáltnak. Legyen $M_{r+1} = M_r$, növeljük r értékét 1-gyel, és folytassuk az eljárást a következő iterációs lépéssel.

- *6. lépés.* (Páratlan élszámú kör.) Növeljük i értékét 1-gyel. A kapott páratlan kört jelölje $\mathcal{C}_i$. Zsugorítsuk a $\mathcal{G}_{i-1}$ gráfot $\mathcal{C}_i$-vel, legyen a $\mathcal{C}_i$-nek megfelelő mesterséges csúcs a_i^*. Jelölje a zsugorított gráfot $\mathcal{G}_i$ és $\bar{M}_i$ az $\bar{M}_{i-1}$ párosítás által indukált párosítást $\mathcal{G}_i$-ben. Töröljük a címkéket a $\mathcal{G}_i$ gráfból. Legyen $s = a_i^*$ és folytassuk az eljárást a 3. lépéssel.

Az algoritmus helyességét nem igazoljuk. Az eljárás végrehajtásának bemutatására a 3.1.2. példa $\mathcal{G}$ gráfját tekintjük és meghatározzuk $\mathcal{G}$-nek egy maximális élszámú párosítását.

Előkészítő rész. $M_0 = \emptyset$, $r = 0$.

Iterációs rész. (0-adik iteráció.)

1. lépés. $i = 0$.
2. lépés. Válasszuk az 1 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \emptyset$, $s = 1$.
3. lépés. Alternáló fa eljárás. Az $(1, 6)$ élet választva, az eredmény az $(1, 6)$ javító út.
4. lépés. $R_0 = \{(1, 6)\}$, $M_1 = \{(1, 6)\}$, $r = 1$.

Iterációs rész. (1. iteráció.)

1. lépés. $i = 0$.

2. lépés. Válasszuk a 2 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \{(1,6)\}$, $s = 2$.
3. lépés. Alternáló fa eljárás. A $(2,3)$ élet választva, az eredmény a $(2,3)$ javító út.
4. lépés. $R_0 = \{(1,6),(2,3)\}$, $M_1 = \{(1,6),(2,3)\}$, $r = 2$.

Iterációs rész. (2. iteráció.)

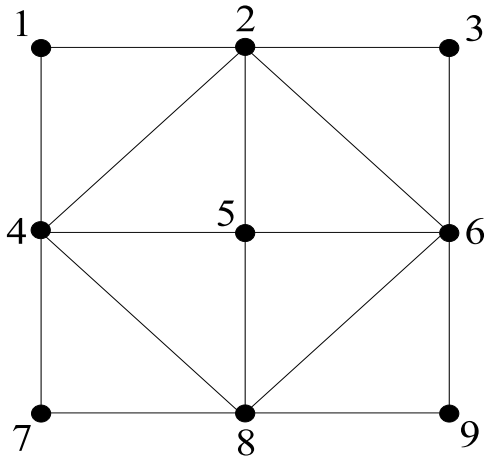
1. lépés. $i = 0$.
2. lépés. Válasszuk az 5 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \{(1,6),(2,3)\}$, $s = 5$.
3. lépés. Alternáló fa eljárás. A $(3,5),(5,6),(1,2)$ éleket választva, az eredmény az $(1,2),(2,3),(3,5),(5,6),(1,6)$ élekből álló kör.
6. lépés. $i = 1$, $\mathcal{C}_1 = \{(1,2),(2,3),(3,5),(5,6),(1,6)\}$, $\bar{M}_1 = \emptyset$, $\mathcal{G}_1 = (\{4, a_1^*\}, \{(4,a_1^*)_{4,5}, (4,a_1^*)_{3,4}\})$, töröljük $\mathcal{G}_1$-ből a címkéket.
3. lépés. Alternáló fa eljárás. A $(4,a_1^*)_{3,4}$ élet választva, az eredmény a $(4,a_1^*)_{3,4}$ javító út.
4. lépés. $R_1 = \{(4,a_1^*)_{3,4}\}$, $R_0 = \{(3,4),(1,2),(5,6)\}$, $M_3 = \{(3,4),(1,2),(5,6)\}$, $r = r + 1 = 3$.

Iterációs rész. (3. iteráció.)

1. lépés. Nincs $\mathcal{G}$-ben M_3-ra nézve szabad csúcs, de akkor nincs átvizsgálatlan szabad csúcs sem, így vége az eljárásnak, M_3 egy maximális élszámú párosítása $\mathcal{G}$-nek.

A következő példában egy olyan esetet mutatunk be, amikor kétszer kell kört zsugorítani.

3.1.3. példa. Tekintsük a $\mathcal{G} = (V, E)$ gráfot, ahol $V = \{1, \ldots, 9\}$ és $E = \{(1,2),(1,4),(2,3),(2,4),(2,5),(2,6),(3,6),(4,5),(4,7),(4,8),(5,6),(5,8),(6,8),(6,9),(7,8),(8,9)\}$. A $\mathcal{G}$ gráf grafikus reprezentációját adja meg a 3.1.3. ábra.



3.1.3. ábra. A 3.1.3. példa $\mathcal{G}$ gráfjának grafikus reprezentációja.

Előkészítő rész. $M_0 = \emptyset$, $r = 0$.

Iterációs rész. (0. iteráció.)

1. lépés. $i = 0$.
2. lépés. Válasszuk az 1 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \emptyset$, $s = 1$.
3. lépés. Alternáló fa eljárás. Az $(1, 2)$ élet választva, az eredmény az $(1, 2)$ javító út.
4. lépés. $R_0 = \{(1, 2)\}$, $M_1 = \{(1, 2)\}$, $r = 1$.

Iterációs rész. (1. iteráció.)

1. lépés. $i = 0$.
2. lépés. Válasszuk a 4 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \{(1, 2)\}$, $s = 4$.
3. lépés. Alternáló fa eljárás. A $(4, 5)$ élet választva, az eredmény a $(4, 5)$ javító út.
4. lépés. $R_0 = \{(1, 2), (4, 5)\}$, $M_2 = \{(1, 2), (4, 5)\}$, $r = 2$.

Iterációs rész. (2. iteráció.)

1. lépés. $i = 0$.
2. lépés. Válasszuk a 6 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \{(1, 2), (4, 5)\}$.
3. lépés. Alternáló fa eljárás. Az $(5, 6), (4, 5), (4, 7)$ éleket választva, az eredmény az $(5, 6), (4, 5), (4, 7)$ javító út.
4. lépés. $R_0 = \{(1, 2), (5, 6), (4, 7)\}$, $M_3 = \{(1, 2), (5, 6), (4, 7)\}$, $r = 3$.

Iterációs rész. (3. iteráció.)

1. lépés. $i = 0$.
2. lépés. Válasszuk a 8 csúcsot. $\mathcal{G}_0 = \mathcal{G}$, $\bar{M}_0 = \{(1, 2), (4, 7), (5, 6)\}$, $s = 8$.
3. lépés. Alternáló fa eljárás. A $(6, 8), (5, 6), (7, 8), (4, 7), (4, 5)$ éleket választva az eredmény a $(4, 5), (5, 6), (6, 8), (7, 8), (4, 7)$ élekből álló kör.
6. lépés. $\mathcal{C}_1 = \{(4, 5), (5, 6), (6, 8), (7, 8), (4, 7)\}$, $\bar{M}_1 = \{(1, 2)\}$, $\mathcal{G}_1 = (V_1, E_1)$, ahol $V_1 = \{1, 2, 3, 9, a_1^*\}$ és $E_1 = \{(1, 2), (1, a_1^*)_{1,4}, (2, 3), (2, a_1^*)_{2,4}, (2, a_1^*)_{2,5}, (2, a_1^*)_{2,6}, (3, a_1^*)_{3,6}, (9, a_1^*)_{6,9}, (9, a_1^*)_{8,9}\}$, $\bar{M}_1 = \{(1, 2)\}$. Töröljük $\mathcal{G}_1$-ből a címkéket.
3. lépés. Alternáló fa eljárás ($\mathcal{G}_1$-re és $\bar{M}_1$-re). A $(2, a_1^*)_{2,4}$ és $(1, a_1^*)_{1,4}$ éleket választva, az eredmény az $(1, 2), (1, a_1^*)_{1,4}, (2, a_1^*)_{2,4}$ élekből álló kör.
6. lépés. $\mathcal{C}_2 = \{(1, 2), (1, a_1^*)_{1,4}, (2, a_1^*)_{2,4}\}$, $\bar{M}_2 = \emptyset$, $\mathcal{G}_2 = (V_2, E_2)$, ahol $V_2 = \{3, 9, a_2^*\}$ és $E_2 = \{(3, a_2^*)_{3,6}, (3, a_2^*)_{2,3}, (9, a_2^*)_{6,9}, (9, a_2^*)_{8,9}\}$. Töröljük $\mathcal{G}_2$-ből a címkéket.
3. lépés. Alternáló fa eljárás ($\mathcal{G}_2$-re és $\bar{M}_2$-re). A $(3, a_2^*)_{2,3}$ élet választva, az eredmény a $(3, a_2^*)_{2,3}$ javító út.
4. lépés. $R_2 = \{(3, a_2^*)_{2,3}\}$, $R_1 = \{(2, 3), (1, a_1^*)_{1,4}\}$, $R_0 = \{(2, 3), (1, 4),$

$$(5,6),(7,8)\},\ M_4 = \{(2,3),(1,4),(5,6),(7,8)\},\ r = 4.$$

Iterációs rész. (4. iteráció.)

1. lépés. Mivel $\mathcal{G}$-ben egyetlen szabad csúcs van M_4-re nézve (a 9 csúcs), ezért M_4 a $\mathcal{G}$ gráf egy maximális számú élet tartalmazó párosítása.

3.2. Minimális súlyú teljes párosítási feladat páros gráfban

A címben szereplő problémát a következőképpen fogalmazhatjuk meg.

Legyen adott egy $\mathcal{G} = (A \cup B, E)$ páros gráf, amelyre $|A| = |B|$ teljesül. Továbbá legyen adott $\mathcal{G}$ minden $(a, b) \in E$ élére egy w_{ab} súly. Feladat $\mathcal{G}$ egy minimális súlyú teljes párosításának meghatározása, azaz keressük a

$$\min\{\textstyle\sum_{(a,b)\in M} w_{ab} : M \ \mathcal{G} \text{ teljes párosítása } \} \tag{3.2.1}$$

feladat egy optimális megoldását, ahol a lehetséges megoldások $\mathcal{G}$ teljes párosításai, és a célfüggvény minden teljes párosításhoz hozzárendeli annak súlyát.

Nyilvánvaló, hogy a (3.2.1) feladatnak nem szükségképp létezik lehetséges megoldása (egy teljes párosítása $\mathcal{G}$-nek), és ekkor optimális megoldás sem létezik. Másrészt, mivel a lehetséges megoldások száma véges, ezért ha (3.2.1)-nek létezik lehetséges megoldása, akkor létezik egy optimális megoldása is. Így, ha a (3.2.1) feladatot akarjuk megoldani, akkor

- el kell dönteni, hogy létezik-e lehetséges megoldása,
- ha létezik lehetséges megoldása, akkor meg kell határozni egy optimális megoldását.

A továbbiakban egy olyan megoldó algoritmussal ismerkedünk meg, amely egy eljárás keretében megoldja az eldöntési problémát, továbbá, ha létezik lehetséges megoldás, akkor szolgáltatja az optimális megoldást is. Mielőtt erre rátérnénk, megadjuk a probléma néhány gyakorlati alkalmazását.

Munkák optimális kiosztása

Adott bizonyos számú dolgozó és ugyanennyi munka. Az egyes dolgozók a munkákat különböző költségekkel tudják végrehajtani. Osszuk szét a dolgozók között az összes munkát úgy, hogy minden dolgozó pontosan egy munkát kapjon, és a munkavégzés összköltsége minimális legyen.

Tegyük fel, hogy $d_1, \ldots, d_n$ jelölik a dolgozókat és $m_1, \ldots, m_n$ a munkákat. Tekintsük a $\mathcal{G} = (V, E)$ irányítatlan gráfot, ahol $V = \{d_1, \ldots, d_n, m_1, \ldots, m_n\}$ és minden d_i, m_j párosra $(d_i, m_j) \in E$ akkor és csak akkor, ha a d_i dolgozó

képes elvégezni az m_j munkát. Minden $(d_i, m_j) \in E$ élre legyen c_{ij} annak a munkavégzésnek a költsége, amikor a d_i dolgozó hajtja végre az m_j munkát, és rendeljük c_{ij}-t a (d_i, m_j) élhez súlyként. Akkor $\mathcal{G}$ egy minimális súlyú teljes párosítása optimális megoldása a fenti feladatnak.

Most megadunk egy további olyan problémát, amely a tekintett párosítási feladathoz vezet. A "házasságkötési játék" néven ismert alábbi feladatot P. R. Halmos és H. Vaughon [84] publikálta 1950-ben.

Házasságkötési játék

Telepesek egy 10 legényemberből álló kolóniája kiegészül 10 potenciális menyasszonnyal. Rövid udvarlás után a fiatalemberek elhatározzák, hogy haladéktalanul megházasodnak. Minden egyes menyasszonyjelölt kap egy névsort, amelyen a 10 fiatalember neve szerepel, és ezen kell pontoznia a férjjelölteket, a számára leginkább megfelelőt 10 ponttal, a következő választottját 9 ponttal, és így tovább. Tegyük fel, hogy bármely rögzített párválasztásnál az egész telep boldogsága a menyasszonyjelöltek által adott pontszámok összegével arányos. Határozzunk meg egy olyan párválasztást, amely az egész telep számára maximális boldogságot eredményez.

A fenti probléma egy olyan párosítási feladattal oldható meg, melynek $\mathcal{G}$ gráfjában a fiúk és a lányok a csúcspontok, minden fiútól vezet él minden lányhoz, és az i-edik fiútól a j-edik lányhoz vezető él súlya $-w_{ij}$, ahol w_{ij} a j-edik lánynak az i-dik fiúra adott pontszámát jelöli. Ekkor $\mathcal{G}$ egy minimális súlyú teljes párosításával szolgáltatott párválasztás az egész telep számára maximális boldogságot eredményez.

A (3.2.1) feladat megoldásában fontos szerepe van az alábbi észrevételnek.

Legyen $W = \max\{|w_{ab}| : (a, b) \in E\}$. Jelölje (3.2.2) azt a párosítási feladatot, amelynek gráfja megegyezik a (3.2.1) feladat gráfjával csak a súlyok mások. Minden $(a, b) \in E$ élre az él súlya legyen $W + w_{ab}$ a (3.2.2) feladatban. Jelölje (3.2.1) célfüggvényét w_1 és (3.2.2) célfüggvényét w_2. Akkor az alábbi állítások nyilvánvalóan teljesülnek.

- a (3.2.1) és (3.2.2) feladatoknak ugyanazok a párosítások a lehetséges megoldásai, azaz a lehetséges megoldások halmaza közös,
- a $\mathcal{G}$ minden M teljes párosítására $w_2(M) = w_1(M) + nW$ teljesül, ahol $n = |A|$, azaz a két célfüggvény a lehetséges megoldások halmazán csak egy konstansban tér el egymástól.

A fenti két tényből közvetlenül adódik az alábbi állítás.

3.2.1. segédtétel. *A (3.2.1) és (3.2.2) feladatoknak egyidejűleg létezik optimális megoldása és az optimális megoldások megegyeznek, azaz ami optimális megoldása az egyik feladatnak az a másiknak is és fordítva.*

A segédtételből és W definíciójából kapjuk a következő észrevételt.

3.2.1. következmény. *Az optimális megoldás létezését és meghatározását illetően elegendő olyan (3.2.1) típusú feladatok vizsgálatára szorítkozni, amelyekben az élek súlyai rendre nemnegatívak.*

A továbbiakban mindig feltesszük, hogy a (3.2.1) feladatban az élek súlyai nemnegatívak.

Vegyük észre, hogy a bemutatott két alkalmazásnál az elsőben a gráf nem szükségképp teljes páros gráf, ugyanis a d_i dolgozót nem kötjük össze éllel az m_j munkával, ha azt nem képes végrehajtani, viszont a másodikban a gráf teljes páros gráf mivel minden fiú éllel össze van kötve minden lánnyal. Nyilvánvalóan nem teljes páros gráf esetén nem szükségképp létezik lehetséges megoldás és így optimális megoldás sem, míg teljes páros gráf esetén mindig létezik optimális megoldás. Most megmutatjuk, hogy a lehetséges megoldás létezésének eldöntése nem teljes páros gráf esetén visszavezethető egy alkalmas teljes páros gráfot tartalmazó párosítási feladat megoldására.

Ehhez tekintsünk egy (3.2.1) típusú párosítási feladatot, amelyben $\mathcal{G} = (A \cup B, E)$ nem teljes páros gráf és az élek w_{ab} súlyai nemnegatívak. Legyen $W \geq 1 + \sum_{(a,b)\in E} w_{ab}$. Konstruáljuk meg a tekintett feladathoz a következő (3.2.3)-mal jelölt párosítási feladatot. A (3.2.3) feladat gráfja legyen a $\mathcal{G}' = (A \cup B, E')$ teljes páros gráf, és $\mathcal{G}'$ éleinek a súlyait definiáljuk a következők szerint. Minden $(a, b) \in E'$ élre legyen az (a, b) él súlya d_{ab}, ahol

$$d_{ab} = \begin{cases} w_{ab}, & \text{ha } (a,b) \in E, \\ W & \text{különben.} \end{cases}$$

A tekintett két feladat közötti kapcsolatot adja meg az alábbi állítás.

3.2.2. segédtétel. *A (3.2.1) feladatnak akkor és csak akkor létezik optimális megoldása, ha a hozzá konstruált (3.2.3) feladat optimuma kisebb, mint W, és ebben az esetben (3.2.3) optimális megoldása egyben optimális megoldása a tekintett (3.2.1) feladatnak is.*

Bizonyítás. Jelölje z_1 a (3.2.1) feladat és z_2 a (3.2.3) feladat célfüggvényét. Tegyük fel, hogy a (3.2.1) feladatnak létezik optimális megoldása.

Jelöljön $M \subseteq E$ egy optimális megoldást. Ekkor mivel M minden éle E-beli és a súlyok nemnegatívak, azt kapjuk, hogy

$$z_2(M) = \sum_{(a,b)\in M} d_{ab} = \sum_{(a,b)\in M} w_{ab} < W.$$

Mivel M lehetséges megoldása a (3.2.3) feladatnak is, a kapott egyenlőtlenségből következik, hogy (3.2.3) optimuma kisebb, mint W.

Most tegyük fel, hogy a (3.2.3) feladat optimuma kisebb, mint W. Jelöljön $M \subseteq E'$ egy optimális megoldást. Akkor

$$z_2(M) = \sum_{(a,b)\in M} d_{ab} < W.$$

Mivel a súlyok nemnegatívak, ebből az egyenlőtlenségből rögtön következik, hogy minden M-beli él E-ben van. De akkor M lehetséges megoldása (3.2.1)-nek, és így (3.2.1)-nek létezik optimális megoldása.

Még azt kell igazolnunk, hogy a másodikként tárgyalt esetben (3.2.3) M optimális megoldása egyben (3.2.1)-nek is optimális megoldása. Ha nem így lenne, akkor (3.2.1)-nek lenne egy olyan M' lehetséges megoldása, amelyre $z_1(M') < z_1(M)$ teljesülne. Mivel M' lehetséges megoldása (3.2.3)-nak is és $z_1(M') = z_2(M')$, $z_1(M) = z_2(M)$, a $z_1(M') < z_1(M)$ egyenlőtlenség ellentmond annak, hogy M a (3.2.3) feladat optimális megoldása.

A 3.2.2. segédtételből adódik az alábbi észrevétel.

3.2.2. következmény. *Az optimális megoldás létezését és meghatározását illetően elegendő olyan (3.2.1) típusú feladatok vizsgálatára szorítkozni, amelyeknek gráfja teljes páros gráf és az élek súlyai rendre nemnegatívak.*

Tekintsünk egy, a 3.2.2. következményben leírtakat teljesítő párosítási feladatot. Legyen a feladat gráfja $\mathcal{G} = (A \cup B, E)$, ahol $|A| = |B| = n$. Az általánosság megszorítása nélkül feltehetjük, hogy $A = \{a_1, \ldots, a_n\}$, $B = \{b_1, \ldots, b_n\}$, és jelölje c_{ij} az (a_i, b_j) él súlyát. Most $\mathcal{G}$ egy M teljes párosítását egy $n \times n$-es bináris mátrixszal írjuk le a következő módon. Legyen

$$x_{ij} = \begin{cases} 1, & \text{ha } (a_i, b_j) \in M, \\ 0 & \text{különben.} \end{cases}$$

Felhasználva a bevezetett jelöléseket, a tekintett problémát a következő bináris lineáris programozási feladattal írhatjuk le.

$$\sum_{t=1}^{n} x_{it} = 1 \ \ (i = 1, \ldots, n)$$

$$\sum_{t=1}^{n} x_{tj} = 1 \ \ (j = 1, \ldots, n) \tag{3.2.4}$$

$$x_{ij} \in \{0, 1\} \ \ (i = 1, \ldots, n; \ j = 1, \ldots, n)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} = z \to \min$$

Valóban, minden i-re a $\sum_{t=1}^{n} x_{it} = 1$, $x_{it} \in \{0, 1\}$ $(t = 1, \ldots, n)$ feltétel biztosítja, hogy az a_i csúcsból pontosan egy él vezet a B halmazba, és minden j-re a $\sum_{t=1}^{n} x_{tj} = 1$, $x_{tj} \in \{0, 1\}$ $(t = 1, \ldots, n)$ feltétel pedig azt eredményezi, hogy a b_j csúcsba pontosan egy A-beli pontból vezet él. Ebből már következik, hogy minden B-beli csúcs pontosan egy élnek a végpontja, és minden A-beli csúcs pontosan egy élnek a végpontja. A megfelelő teljes párosítás súlya c_{ij} definíciója alapján: $\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$.

Mielőtt nekilátnánk a (3.2.4) feladat megoldásának, fontosnak tartjuk megjegyezni a következőket. A (3.2.4) optimumszámítási modell a szakirodalomban más néven is ismeretes, nevezetesen *hozzárendelési feladatnak* nevezik. Ez a modell már az 50-es évek elején a vizsgálatok tárgyát képezte, míg a párosítási feladatok vizsgálatának kezdete az 50-es évek végére tehető, amikoris C. Berge [21] bevezette és alkalmazta a párosítási feladatok megoldásában kulcsszerepet játszó alternáló utakat, és igazolta a javító utakra vonatkozó tételt. Tekintettel erre a függetlenségre a továbbiakban a (3.2.4) feladatot mi is hozzárendelési feladatnak fogjuk nevezni.

Vizsgáljuk ezek után a (3.2.4) feladatot. Vegyük észre, hogy a tekintett feladat lehetséges megoldásai olyan 0 és 1 elemekből álló $n \times n$-es mátrixok, amelyeknek minden sora és minden oszlopa pontosan egy darab 1-est tartalmaz. Igaz ennek a megfordítása is, azaz minden ilyen mátrix lehetséges megoldása (3.2.4)-nek. Másrészt könnyen belátható, hogy az ilyen tulajdonságú mátrixok száma $n!$. Az elmondottakból következik, hogy rögzített n mellett a különböző célfüggvényű hozzárendelési feladatokhoz egy közös lehetséges megoldás halmaz tartozik, mégpedig a fenti tulajdonsággal rendelkező mátrixok halmaza. Így a (3.2.4) feladatot egyértelműen meghatározza a (c_{ij}) mátrix, amit *súlymátrixnak* vagy *költségmátrixnak* nevezünk.

Ez a meghatározottság lehetővé teszi, hogy tetszőleges $\mathbf{D}^{n\times n}$ mátrixra, a $\mathbf{D}$ súlymátrixú hozzárendelési feladatot $A(\mathbf{D})$-vel jelöljük az angol Assignment Problem elnevezés után.

A lehetséges megoldások halmazának az előzőekben adott leírását felhasználva, megkaphatjuk a feladat optimális megoldását úgy, hogy valamilyen eljárással előállítjuk a fenti tulajdonságú mátrixok mindegyikét, majd kiszámítjuk a mátrixokhoz tartozó célfüggvényértékeket és vesszük ezek minimumát. Ennek az eljárásnak a bemutatására tekintsük az

$$A\begin{pmatrix}3 & 4 & 2\\ 6 & 5 & 1\\ 7 & 4 & 3\end{pmatrix} \qquad \text{hozzárendelési feladatot.}$$

A lehetséges megoldások halmaza a következő mátrixokból áll:

$$\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 1 & 0\end{pmatrix}\begin{pmatrix}0 & 1 & 0\\ 1 & 0 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 1 & 0 & 0\end{pmatrix}\begin{pmatrix}0 & 0 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}\begin{pmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{pmatrix}$$

A fenti mátrixokhoz tartozó célfüggvényértékek rendre: 11, 8, 13, 12, 12, 14. Így a feladat optimális megoldása:

$$\bar{\mathbf{X}} = \begin{pmatrix}1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 1 & 0\end{pmatrix}\ , \qquad \text{az optimum értéke pedig } z(\bar{\mathbf{X}}) = 8.$$

Nyilvánvaló, hogy nagyobb feladatok esetén ez az eljárás gyakorlatilag nem használható, ugyanis nagyon nagy lesz az előállítandó mátrixok száma. Például, ha $n = 6$, akkor a megadott tulajdonságú mátrixok száma: $6! = 720$. Ezért a (3.2.4) feladat megoldására egy másik, a fenti algoritmusnál lényegesen hatékonyabb eljárással fogunk megismerkedni, amely *magyar módszer* néven ismeretes.

Az eljárást H. W. Kuhn publikálta 1955-ben. A módszer felfedezéséről Kuhn a Szigma folyóiratban [118] írt egy igen érdekes és tanulságos visszaemlékezést. Az elbeszélésben Kuhn leírja a kutatás körülményeit, annak egyes lépéseit. A történet 1953 nyarán kezdődik, amikor a Nemzeti Mérésügyi Hivatal és más amerikai kormányzati szervek összegyűjtöttek egy kiváló kombinatorikus és algebrista szakemberekből álló csoportot a Kaliforniai Egyetemen. Az intézetben volt az egyik legjobb korabeli számítógép, a Standard

Western Automatic Computer (SWAC), amelynek az egész memóriája 256 darab Williamson katódcsőből állt.

A nyár folyamán Kuhn egyik kollegája, C. B. Tompkins 10×10-es hozzárendelési feladatokat próbált a (SWAC) segítségével megoldani az összes $10! = 3.628.800$ permutáció leszámlálásával. Tompkins egyetlen próbálkozása sem járt sikerrel.

Kuhn ebben az időszakban Kőnig klasszikus gráfelméleti könyvét [113] olvasta és rájött, hogy egy gráf két n szögpontú részre particionálása és a két rész közötti párosítás problémája pontosan megegyezik egy olyan $n \times n$-es hozzárendelési feladattal, amelyben a költségmátrix elemei binárisak. De ami még ennél is fontosabb, Kuhn észrevette, hogy Kőnig megadott egy olyan algoritmust, amellyel meghatározható volt a fenti párosítási probléma optimális megoldása. Az algoritmus a következő eredményen alapul: Legyen $\mathbf{C}$ $n \times n$-es bináris mátrix. Fedjük le $\mathbf{C}$ bizonyos sorait és oszlopait. Ha ezek a fedések a $\mathbf{C}$ mátrixban található összes 1-est lefedik, akkor *fedőrendszerről* beszélünk. A legkevesebb sor, illetve oszlop fedést használó fedőrendszert *minimális fedésnek* nevezzük. Válasszunk $\mathbf{C}$-ből maximális számú 1-est úgy, hogy egy sorból vagy oszlopból legfeljebb egy 1-est választhatunk. Az ilyen módon választható 1-esek maximális száma megegyezik a $\mathbf{C}$ minimális fedésében szereplő sor és oszlop fedések számával. Ezek után már csak a következő probléma megoldása maradt hátra: hogyan lehet az általános hozzárendelési feladatot olyan hozzárendelési feladatra visszavezetni, amelyben a költségmátrix bináris. Figyelmesebben olvasva Kőnig könyvét, Kuhn felfigyelt egy lábjegyzetre, amely Egerváry [50] cikkére utalt, és ráérzett, hogy a fenti probléma megoldásának kulcsa talán éppen Egerváry cikkében található meg. Amikor ősszel visszatért a Bryn Mawr College-ba, kikölcsönözte a College-ből Egerváry cikkének egy másolatát egy nagy magyar szótár és nyelvtan kiséretében. Két hétig tanult magyarul és közben lefordította Egerváry cikkét [51]. Elképzelésének megfelelően Egerváry cikke tartalmazott egy módszert, melynek alapján az általános hozzárendelési feladat visszavezethető véges sok bináris költségmátrixú feladatra. Felhasználva Egerváry redukciós eljárását és Kőnig maximum-párosítási algoritmusát, 1953 őszén számos 12×12-es hozzárendelési feladat optimális megoldását számította ki kézzel. Mindegyik feladatot két órán belül meg tudta oldani és ez meggyőzte arról, hogy ez a kombinált algoritmus jó. A történet ezen részéhez Kuhn az alábbi kedves észrevételt tette:

Valószínűleg ez egyike volt azon legutolsó eseteknek, amikor papírral és tollal le lehetett győzni a világ legnagyobb és leggyorsabb elektronikus számítógépét.

Abból a célból, hogy teljesen nyilvánvaló legyen, miszerint az általa javasolt algoritmust két magyar matematikus, Kőnig és Egerváry munkája inspirálta, Kuhn az eljárást "magyar módszer"-nek nevezte el és ezen a néven is publikálta (ld. [116], [117]).

Az 1955-ös publikáció után a magyar módszer igen ismertté vált, és különböző formákban került tárgyalásra. Mi a továbbiakban D. Yudin, és E. G. Gol'shtein 1969-es könyvére [181] építve tárgyaljuk az eljárást. Mielőtt erre rátérnénk, szükségesek bizonyos előkészületek.

Legyenek $\mathbf{C}^{n\times m}$ és $\mathbf{D}^{n\times m}$ tetszőleges valós mátrixok. Azt mondjuk, hogy $\mathbf{C}$ *ekvivalens* $\mathbf{D}$-vel, ha vannak olyan $\alpha_1,\ldots,\alpha_n;\ \beta_1,\ldots,\beta_m$ valós számok, hogy bármely $1\le i\le n$, $1\le j\le m$ indexpárra $c_{ij}=d_{ij}+\alpha_i+\beta_j$ teljesül.

A bevezetett relációra a továbbiakban a $\mathbf{C}\sim\mathbf{D}$ jelölést fogjuk használni. Nyilvánvaló, hogy az azonos típusú valós mátrixok halmazán ez a reláció reflexív, tranzitív és szimmetrikus, azaz ekvivalenciareláció.

A hozzárendelési feladat és a fenti reláció kapcsolatát mutatja a következő állítás.

3.2.3. segédtétel. *Ha* $\mathbf{C}$, $\mathbf{D}$ $n\times n$*-es valós mátrixok és* $\mathbf{C}\sim\mathbf{D}$, *akkor az* $A(\mathbf{C})$ *és* $A(\mathbf{D})$ *hozzárendelési feladatok optimális megoldásai megegyeznek.*

Bizonyítás. Mivel a lehetséges megoldások halmaza közös, és mindkét feladatnak létezik optimális megoldása, ezért az állítás korrekt. Ezek után jelölje $A(\mathbf{C})$ és $A(\mathbf{D})$ célfüggvényét rendre $z_{\mathbf{C}}$ és $z_{\mathbf{D}}$. Megmutatjuk, hogy van olyan γ konstans, amelyre $z_{\mathbf{C}}(\bar{\mathbf{X}})=z_{\mathbf{D}}(\bar{\mathbf{X}})+\gamma$ teljesül minden $\bar{\mathbf{X}}$ lehetséges megoldásra. Valóban, mivel $\mathbf{C}\sim\mathbf{D}$, így vannak olyan $\alpha_1,\ldots,\alpha_n;\ \beta_1,\ldots,\beta_n$ konstansok, hogy $c_{ij}=d_{ij}+\alpha_i+\beta_j$ teljesül bármely $1\le i\le n$, $1\le j\le n$ indexpárra. De akkor

$$z_{\mathbf{C}}(\bar{\mathbf{X}})=\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}\bar{x}_{ij}=\sum_{i=1}^{n}\sum_{j=1}^{n}(d_{ij}+\alpha_i+\beta_j)\bar{x}_{ij}=$$
$$\sum_{i=1}^{n}\sum_{j=1}^{n}d_{ij}\bar{x}_{ij}+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}\bar{x}_{ij}+\sum_{j=1}^{n}\beta_j\sum_{i=1}^{n}\bar{x}_{ij}\ .$$

Mivel $\bar{\mathbf{X}}$ lehetséges megoldás, ezért $\sum_{j=1}^{n}\bar{x}_{ij}=1$ és $\sum_{i=1}^{n}\bar{x}_{ij}=1$. De akkor

$$z_{\mathbf{C}}(\bar{\mathbf{X}})=z_{\mathbf{D}}(\bar{\mathbf{X}})+\sum_{i=1}^{n}\alpha_i+\sum_{j=1}^{n}\beta_j\ .$$

Most legyen $\gamma=\sum_{i=1}^{n}\alpha_i+\sum_{j=1}^{n}\beta_j$. Akkor a $z_{\mathbf{C}}(\bar{\mathbf{X}})=z_{\mathbf{D}}(\bar{\mathbf{X}})+\gamma$ egyenlethez jutunk, amely teljesül bármely $\bar{\mathbf{X}}$ lehetséges megoldásra. Ez pontosan

azt jelenti, hogy a lehetséges megoldások halmazán a két célfüggvény csak egy additív konstansban tér el egymástól, amiből nyilvánvalóan következik az állítás.

A továbbiakban alapvető szerepe lesz a következő definíciónak:

Adott mátrix 0 elemeinek egy rendszerét *független 0-rendszernek* nevezzük, ha a mátrix minden sora és minden oszlopa legfeljebb egy elemét tartalmazza a rendszernek. Egy független 0-rendszer elemeit a továbbiakban 0^*-gal fogjuk jelölni.

Tekintsük a következő független 0-rendszereket:

$$\begin{pmatrix} 0^* & 1 & 2 & 3 \\ 4 & 2 & 0 & 0 \\ 3 & 4 & 0 & 0 \\ 0 & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 2 & 3 \\ 4 & 2 & 0^* & 0 \\ 3 & 4 & 0 & 0^* \\ 0 & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} 0^* & 1 & 2 & 3 \\ 4 & 2 & 0^* & 0 \\ 3 & 4 & 0 & 0^* \\ 0 & 0^* & 1 & 2 \end{pmatrix} \begin{pmatrix} 0^* & 1 & 2 & 3 \\ 4 & 2 & 0 & 0^* \\ 3 & 4 & 0^* & 0 \\ 0 & 0^* & 1 & 2 \end{pmatrix}$$

A fenti példák mutatják, hogy adott mátrixban több különböző független 0-rendszer is kijelölhető, és maximális elemszámú független 0-rendszerből lehet több is.

A 0^*-okon kívül használni fogjuk még a következő elnevezéseket, illetve jelöléseket. Egy mátrix valamely sorát (oszlopát) *kötött sornak* (*oszlopnak*) nevezzük, ha mellette (felette) egy "+" jel áll. A mátrix valamely elemét *szabad elemnek* nevezzük, ha nincs semmiféle jellel ellátva, és sem a sora, sem az oszlopa nincsen lekötve. Speciálisan, ha az illető elem 0, akkor *szabad 0*-ról beszélünk. Végül használni fogjuk a $\mathbf{C} \geq 0$ jelölést, ha a $\mathbf{C}$ mátrix minden eleme nemnegatív.

Most rátérhetünk az $A(\mathbf{C}^{n\times n})$ hozzárendelési feladat magyar módszerrel történő megoldására. Az eljárással előállítunk egy olyan $\mathbf{C}^{(0)}, \mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(k)}$ $(k < n)$ mátrixsorozatot, amelyre teljesülnek a következők:

(1) $\mathbf{C} \sim \mathbf{C}^{(0)}$,

(2) $\mathbf{C}^{(t)} \sim \mathbf{C}^{(t+1)}$ $(t = 0, \ldots, k-1)$,

(3) $\mathbf{C}^{(t)} \geq 0$ $(t = 0, \ldots, k)$,

(4) $\mathbf{C}^{(k)}$-ban ki van jelölve egy n-elemű független 0-rendszer.

Tegyük fel, hogy rendelkezünk egy, a fentieket kielégítő $\mathbf{C}^{(0)}, \mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(k)}$ mátrixsorozattal. Definiáljuk az $\bar{\mathbf{X}}$ mátrixot a következőképpen:

$$\bar{x}_{ij} = \begin{cases} 1, & \text{ha } c_{ij}^{(k)} = 0^*, \\ 0 & \text{különben.} \end{cases}$$

Igazoljuk, hogy $\bar{\mathbf{X}}$ optimális megoldása $A(\mathbf{C})$-nek. Valóban, $\bar{\mathbf{X}}$ lehetséges megoldás, mivel $\mathbf{C}^{(k)}$-ban minden sor és minden oszlop legfeljebb egy 0^*-ot tartalmaz, továbbá a 0^*-ok száma n. Másrészt jelölje z_k az $A(\mathbf{C}^{(k)})$ feladat célfüggvényét. Akkor (3) alapján $z_k(\tilde{\mathbf{X}}) \geq 0$ teljesül tetszőleges $\tilde{\mathbf{X}}$ lehetséges megoldásra. Most vegyük észre, hogy $\bar{\mathbf{X}}$ definíciójából $z_k(\bar{\mathbf{X}}) = 0$ következik. De ekkor $z_k(\tilde{\mathbf{X}}) \geq z_k(\bar{\mathbf{X}})$ teljesül tetszőleges $\tilde{\mathbf{X}}$ lehetséges megoldásra, azaz $\bar{\mathbf{X}}$ optimális megoldása $A(\mathbf{C}^{(k)})$-nak. Végül (1) és (2), valamint az ekvivalencia tranzitivitása alapján $\mathbf{C} \sim \mathbf{C}^{(k)}$. Ebből viszont már a 3.2.3. segédtétellel azt kapjuk, hogy $\bar{\mathbf{X}}$ optimális megoldása $A(\mathbf{C})$-nek, amivel az $\bar{\mathbf{X}}$-ra vonatkozó állítást igazoltuk.

A kérdés ezek után az, hogy miként lehet egy, a fenti tulajdonságokat kielégítő $\mathbf{C}^{(0)}, \ldots, \mathbf{C}^{(k)}$ mátrixsorozatot előállítani. Erre szolgál a következő iterációs eljárás.

Magyar módszer [116]

Előkészítő rész

A $\mathbf{C}$ mátrix i-edik sorából rendre vonjuk ki az i-edik sor elemeinek a minimumát ($i = 1, \ldots, n$), az előálló mátrixot jelölje $\bar{\mathbf{C}}$. A $\bar{\mathbf{C}}$ mátrix j-edik oszlopának elemeiből rendre vonjuk ki az illető oszlop elemeinek a minimumát ($j = 1, \ldots, n$), és a kapott mátrixot jelölje $\mathbf{C}^{(0)}$. Jelöljünk ki $\mathbf{C}^{(0)}$-ban egy független 0-rendszert oszlopfolytonosan, azaz oszloponként haladva, a tekintett oszlopból választható 0-k közül mindig a legkisebb sorindexű 0-t vegyük fel a független 0-rendszerbe. Ezek után legyen r értéke 0, és folytassuk az eljárást az iterációs eljárásrésszel.

Iterációs rész (r. iteráció)

- *1. lépés.* Ha a $\mathbf{C}^{(r)}$-ben kijelölt független 0-rendszer elemeinek a száma n, akkor vége az eljárásnak. Ellenkező esetben kössük le $\mathbf{C}^{(r)}$-ben a 0^*-okat tartalmazó oszlopokat, és folytassuk az eljárást a 2. lépéssel.

- *2. lépés.* Keressünk sorfolytonosan szabad 0-t. Ha nincs szabad 0, akkor az 5. lépés következik. Ha találunk szabad 0-t, akkor vizsgáljuk

meg az illető 0 sorát. Ha ez a sor nem tartalmaz 0^*-ot, akkor a 4. lépés következik, ellenkező esetben a 3. lépéssel folytatjuk az eljárást.

- *3. lépés.* A tekintett szabad 0-t lássuk el ","-vel, kössük le a sorát, és szüntessük meg a sorában lévő 0^* oszlopának lekötését, majd folytassuk az eljárást a 2. lépéssel.

- *4. lépés.* A tekintett szabad 0-t lássuk el ","-vel, és ebből a $0'$-ből kiindulva, képezzünk láncot a következők szerint: minden láncbeli $0'$-re az oszlopában lévő 0^*-gal folytatódik a lánc, és minden láncbeli 0^*-ra a sorában lévő $0'$-vel folytatódik a lánc, feltéve, hogy vannak ilyen elemek. Ellenkező esetben a láncképzés véget ér. Ezek után legyen $\mathbf{C}^{(r+1)}$ a jelölések nélküli aktuális $\mathbf{C}^{(r)}$ mátrix, és lássuk el " $*$ "-gal a $c_{ij}^{(r+1)}$ elemet, ha $c_{ij}^{(r)} = 0^*$ és $c_{ij}^{(r)}$ nem szerepel a láncban, vagy $c_{ij}^{(r)} = 0'$ és $c_{ij}^{(r)}$ eleme a láncnak. Ezek után növeljük r értékét 1-gyel, és folytassuk az eljárást a következő iterációs lépéssel.

- *5. lépés.* Képezzük a szabad elemek minimumát, majd ezt a minimumot vonjuk ki az összes szabad elemből és adjuk hozzá a kétszer kötött (soruk és oszlopuk is le van kötve) elemekhez. Ezt követően az átalakított mátrixot tekintve aktuális $\mathbf{C}^{(r)}$ mátrixnak, folytassuk az eljárást a 2. lépéssel.

Az előzőek alapján az eljárás helyességének bizonyításához azt kell igazolnunk, hogy az eljárás révén előállítható egy olyan $\mathbf{C}^{(0)}, \ldots, \mathbf{C}^{(k)}$ mátrixsorozat, amely rendelkezik az (1)-(4) alatti tulajdonságokkal. Mielőtt erre rátérnénk, részletesen végrehajtjuk az algoritmust egy konkrét feladaton.

3.2.1. példa Tekintsük az $A(\mathbf{C})$ hozzárendelési feladatot, ahol

$$\mathbf{C} = \begin{pmatrix} 4 & 5 & 3 & 2 & 3 \\ 3 & 2 & 4 & 3 & 4 \\ 3 & 3 & 4 & 4 & 3 \\ 2 & 4 & 3 & 2 & 4 \\ 2 & 1 & 3 & 4 & 3 \end{pmatrix}$$

A sorminimumok a következők: $2, 2, 3, 2, 1$. Ezeket rendre kivonva az első,..., ötödik sor elemeiből, a következő $\bar{\mathbf{C}}$ mátrixhoz jutunk.

$$\bar{\mathbf{C}} = \begin{pmatrix} 2 & 3 & 1 & 0 & 1 \\ 1 & 0 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 & 2 \\ 1 & 0 & 2 & 3 & 2 \end{pmatrix}$$

A $\bar{\mathbf{C}}$ mátrix oszlopaira képezve a minimumokat, a következő értékeket kapjuk: $0, 0, 1, 0, 0$. Rendre kivonva ezeket a megfelelő oszlopok elemeiből (most csak a harmadik oszlop elemeiből kell kivonnunk 1-et), az alábbi $\mathbf{C}^{(0)}$ mátrixot kapjuk.

$$\mathbf{C}^{(0)} = \begin{pmatrix} 2 & 3 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix}$$

Most jelöljünk ki $\mathbf{C}^{(0)}$-ban egy független 0-rendszert oszlopfolytonosan. Az első oszlop első 0 eleme $c_{31}^{(0)}$. Vegyük fel ezt az elemet a független 0-rendszerbe, azaz lássuk el " $*$ "-gal, majd térjünk át a második oszlop 0 elemeinek a vizsgálatára. Ebben az oszlopban az első 0 elem $c_{22}^{(0)}$. Ennek a sora nem tartalmaz 0^* elemet, így bővíthetjük vele a rendszert, és áttérünk a harmadik oszlopra. Itt az első 0 elem $c_{13}^{(0)}$, amellyel szintén bővíthető a független 0-rendszer. A negyedik oszlop első 0 eleme $c_{14}^{(0)}$, amelynek a sora tartalmaz 0^*-ot, így ezzel nem bővíthető a rendszer. A negyedik oszlop második 0 eleme $c_{44}^{(0)}$, amivel ismét bővíthető a független 0-rendszer. Végül az utolsó oszlop első 0 eleme $c_{35}^{(0)}$. A harmadik sor már tartalmaz 0^*-ot, így ezzel az elemmel nem bővíthetünk. Mivel az utolsó oszlop több 0 elemet nem tartalmaz, ezért a független 0-rendszer kijelölését, és egyben az eljárás előkészítő részét is befejeztük. A kijelölt független 0-rendszer a következő:

$$\mathbf{C}^{(0)} = \begin{pmatrix} 2 & 3 & 0^* & 0 & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix}$$

Térjünk rá ezek után az iterációs rész végrehajtására. Mivel a független 0-rendszer elemeinek a száma 4, ezért minden olyan oszlopot kössünk le, amely 0^*-ot tartalmaz.

$$\mathbf{C}^{(0)} = \begin{array}{c} \begin{array}{ccccc} + & + & + & + & \end{array} \\ \begin{pmatrix} 2 & 3 & 0^* & 0 & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix} \end{array}$$

Sorfolytonosan szabad 0-t keresve, $c_{35}^{(0)}$ az első szabad 0. A harmadik sor tartalmaz 0^*-ot, így a harmadik lépés szerint $c_{35}^{(0)}$-t ellátjuk ”,”-vel, a harmadik sort lekötjük, majd a sorban lévő 0^* oszlopát (első oszlop) feloldjuk. A kapott mátrix a következő:

$$\mathbf{C}^{(0)} = \begin{array}{c} \begin{array}{ccccc} & + & + & + & \end{array} \\ \begin{pmatrix} 2 & 3 & 0^* & 0 & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0' \\ 0 & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix} \end{array} \begin{array}{l} \\ \\ + \\ \\ \\ \end{array}$$

Ezt követően a 2. lépéssel, sorfolytonos 0 kereséssel folytatjuk az eljárást. Most $c_{41}^{(0)}$ az első szabad 0. A negyedik sor tartalmaz 0^*-ot, ezért a 3. lépés alapján $c_{41}^{(0)}$-t ellátjuk ”,”-vel, a negyedik sort lekötjük, és a negyedik oszlopot feloldjuk.

$$\mathbf{C}^{(0)} = \begin{array}{c} \begin{array}{ccccc} & + & + & & \end{array} \\ \begin{pmatrix} 2 & 3 & 0^* & 0 & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0' \\ 0' & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix} \end{array} \begin{array}{l} \\ \\ + \\ + \\ \\ \end{array}$$

Ismét rátérve a 2. lépésre, $c_{14}^{(0)}$ az első szabad 0. Az első sor tartalmaz 0^*-ot, ezért $c_{14}^{(0)}$-t ”,”-vel látjuk el, az első sort lekötjük, és a harmadik oszlopot feloldjuk.

$$\mathbf{C}^{(0)} = \begin{array}{c} \begin{array}{ccccc} & + & & & \end{array} \\ \begin{pmatrix} 2 & 3 & 0^* & 0' & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0' \\ 0' & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix} \end{array} \begin{array}{l} + \\ \\ + \\ + \\ \\ \end{array}$$

Újra a 2. lépéssel folytatva az eljárást, azt kapjuk, hogy nincs szabad 0. Ekkor az 5. lépés következik. Abból a célból, hogy az 5. lépés végrehajtását megkönnyítsük, fedjük le a kötött sorokat és oszlopokat. Így a le nem fedett elemek lesznek pontosan a szabad elemek. Ezek minimuma 1. Ezt a minimumot kivonva a szabad elemekből és hozzáadva a kétszer fedett elemekhez, a következő mátrixhoz jutunk.

$$\mathbf{C}^{(0)} = \begin{array}{c} \begin{array}{ccccc} & + & & & \end{array} \\ \begin{pmatrix} 2 & 4 & 0^* & 0' & 1 \\ 0 & 0^* & 0 & 0 & 1 \\ 0^* & 1 & 0 & 1 & 0' \\ 0' & 3 & 0 & 0^* & 2 \\ 0 & 0 & 0 & 2 & 1 \end{pmatrix} \end{array} \begin{array}{l} + \\ \\ + \\ + \\ \end{array}$$

Most ismét a 2. lépéssel kell folytatnunk az eljárást. Sorfolytonosan $c_{21}^{(0)}$ az első szabad 0. Mivel $c_{21}^{(0)}$ sora tartalmaz 0^*-ot, ezért $c_{21}^{(0)}$-t ellátjuk ","-vel, a sorát lekötjük, a második oszlopot feloldjuk.

$$\mathbf{C}^{(0)} = \begin{pmatrix} 2 & 4 & 0^* & 0' & 1 \\ 0' & 0^* & 0 & 0 & 1 \\ 0^* & 1 & 0 & 1 & 0' \\ 0' & 3 & 0 & 0^* & 2 \\ 0 & 0 & 0 & 2 & 1 \end{pmatrix} \begin{array}{l} + \\ + \\ + \\ + \\ \end{array}$$

Újra szabad 0-t keresve, $c_{51}^{(0)}$ lesz az első szabad 0. Mivel $c_{51}^{(0)}$ sora nem tartalmaz 0^*-ot, ezért a 4. lépéssel folytatódik az eljárás. $c_{51}^{(0)}$-t ellátjuk ","-vel, és ez lesz a lánc kezdő eleme. A lánc következő eleme az első oszlopban lévő 0^*, azaz $c_{31}^{(0)}$. Ezt az elemet a láncban a harmadik sorban elhelyezkedő $0'$, azaz $c_{35}^{(0)}$ követi. Mivel az ötödik oszlop nem tartalmaz 0^*-ot, ezért $c_{35}^{(0)}$-val a lánc véget ér. Ezek után törölve a jelöléseket, és " $*$ "-gal ellátva a láncon kívüli 0^*-oknak megfelelő elemeket $(c_{13}^{(0)}, c_{22}^{(0)}, c_{44}^{(0)})$, valamint " $*$ "-gal ellátva a láncbeli $0'$-knek megfelelő elemeket $(c_{35}^{(0)}, c_{51}^{(0)})$, az alábbi $\mathbf{C}^{(1)}$ mátrixot kapjuk.

$$\mathbf{C}^{(1)} = \begin{pmatrix} 2 & 4 & 0^* & 0 & 1 \\ 0 & 0^* & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0^* \\ 0 & 3 & 0 & 0^* & 2 \\ 0^* & 0 & 0 & 2 & 1 \end{pmatrix}$$

Mivel $\mathbf{C}^{(1)}$-ben egy 5-elemű független 0-rendszer van kijelölve, ezért az eljárás véget ér. Az előzőek alapján tudjuk, hogy ebben az esetben egy optimális megoldás az alábbi $\bar{\mathbf{X}}$ mátrix:

$$\bar{\mathbf{X}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

A végrehajtott eljárással kapcsolatban fontosnak tartjuk megjegyezni a következőket.

A szükséges iterációk k száma $\mathbf{C}$-től függően a $0, 1, \ldots, n-1$ értékek bármelyike lehet. Például konstruálható olyan 5×5-ös $\mathbf{C}$ mátrix, amelyre alkalmazva az eljárást, $\mathbf{C}^{(0)}$, $\mathbf{C}^{(1)}$, $\mathbf{C}^{(2)}$, $\mathbf{C}^{(3)}$, $\mathbf{C}^{(4)}$ az előálló mátrixsorozat.

A láncképzés során előfordulhat, hogy a lánc egyetlen $0'$ elemből, a kezdő elemből áll. Ilyenkor *elfajuló láncról* beszélünk, amely úgy interpretálható, hogy a benne szereplő $0'$-vel közvetlenül bővíthető az előzőekben előállított független 0-rendszer.

Az algoritmus fenti végrehajtásában az egyes lépéseket szándékosan túlrészleteztük. A $\mathbf{C}^{(0)}$ ismételt kiírása nélkül kijelölhető $\mathbf{C}^{(0)}$-ban a független 0-rendszer, leköthetők az oszlopok, és végrehajthatók a sorkötések valamint az oszlopfeloldások. Ez utóbbit a "+" jel bekarikázásával jelöljük. A mátrix ismételt kiírása csak az 5. lépés végrehajtása után szükséges. Az elmondottaknak megfelelően végrehajtva az eljárást, az alábbi mátrixokhoz jutunk:

$$\begin{pmatrix} 4 & 5 & 3 & 2 & 3 \\ 3 & 2 & 4 & 3 & 4 \\ 3 & 3 & 4 & 4 & 3 \\ 2 & 4 & 3 & 2 & 4 \\ 2 & 1 & 3 & 4 & 3 \end{pmatrix} \quad \begin{pmatrix} 2 & 3 & 1 & 0 & 1 \\ 1 & 0 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 & 2 \\ 1 & 0 & 2 & 3 & 2 \end{pmatrix} \quad \begin{matrix} \oplus & + & \oplus & \oplus & \\ \begin{pmatrix} 2 & 3 & 0^* & 0' & 1 \\ 1 & 0^* & 1 & 1 & 2 \\ 0^* & 0 & 0 & 1 & 0' \\ 0' & 2 & 0 & 0^* & 2 \\ 1 & 0 & 1 & 3 & 2 \end{pmatrix} & \begin{matrix} + \\ \\ + \\ + \\ \end{matrix} \end{matrix}$$

$$\begin{matrix} & \oplus & & & \\ \begin{pmatrix} 2 & 4 & 0^* & 0' & 1 \\ 0' & 0^* & 0 & 0 & 1 \\ 0^* & 1 & 0 & 1 & 0' \\ 0' & 3 & 0 & 0^* & 2 \\ 0' & 0 & 0 & 2 & 1 \end{pmatrix} & \begin{matrix} + \\ + \\ + \\ + \\ \end{matrix} & \begin{pmatrix} 2 & 4 & 0^* & 0 & 1 \\ 0 & 0^* & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0^* \\ 0 & 3 & 0 & 0^* & 2 \\ 0^* & 0 & 0 & 2 & 1 \end{pmatrix} \end{matrix}$$

Ezek után rátérhetünk az eljárás helyességének az igazolására. Megmutatjuk, hogy az algoritmus egy olyan $\mathbf{C}^{(0)}, \ldots, \mathbf{C}^{(k)}$ $(k < n)$ mátrixsorozatot állít elő, amely rendelkezik az $(1) - (4)$ alatti tulajdonságokkal.

Vizsgáljuk elsőként az eljárás előkészítő részét. Ha az i-edik sor elemeinek a minimumát α_i-vel jelöljük ($i = 1, \ldots, n$), akkor nyilvánvaló, hogy $c_{ij} = \bar{c}_{ij} + \alpha_i$ teljesül bármely (i, j) indexpárra. Hasonlóan, a $\bar{\mathbf{C}}$ mátrix j-edik oszlopának minimumát β_j-vel jelölve ($j = 1, \ldots, n$), $\bar{c}_{ij} = c_{ij}^{(0)} + \beta_j$. De akkor $c_{ij} = c_{ij}^{(0)} + \alpha_i + \beta_j$ teljesül bármely (i, j) indexpárra, és így $\mathbf{C} \sim \mathbf{C}^{(0)}$. Másrészt vegyük észre, hogy mivel a sorokból és oszlopokból mindig az illető sorok és oszlopok elemeinek minimumát vonjuk ki, ezért $\mathbf{C}^{(0)} \geq 0$. Szintén a minimumok kivonása miatt $\mathbf{C}^{(0)}$ minden sora és minden oszlopa tartalmaz legalább egy 0-t. Ebből következik, hogy az oszlopfolytonosan kijelölhető független 0-rendszer legalább egy elemet tartalmaz. Ezzel az előkészítő rész helyességét igazoltuk.

Tekintsük ezek után az iterációs eljárásrészt. Tegyük fel, hogy adott egy olyan $\mathbf{C}^{(r)}$ mátrix, amelyre $\mathbf{C} \sim \mathbf{C}^{(r)}$, $\mathbf{C}^{(r)} \geq 0$, és $\mathbf{C}^{(r)}$-ben ki van tüntetve egy m-elemű független 0-rendszer, ahol $1 \leq m < n$.

Elsőként megmutatjuk, hogy a tekintett eljárásrész véges lépésben véget ér. Valóban, vegyük észre, hogy a $2 - 3$ lépéskombináció legfeljebb m-szer hajtódik végre, ugyanis minden egyes végrehajtásnál lekötünk egy sort és feloldunk egy oszlopot. Mivel a kötött oszlopok száma m, ezért ezt a lépéskombinációt m-szer végrehajtva, minden egyes 0^* kötött sorban lesz, így az eljárás vagy a 4. lépéssel, vagy az 5. lépéssel folytatódik. Az 5. lépés szintén legfeljebb m-szer kerül végrehajtásra. Erre a lépésre akkor kerül sor, ha nem találunk szabad 0-t. Mivel a kötött sorok száma legfeljebb $m(< n)$, ezért az aktuális $\mathbf{C}^{(r)}$-ben van kötetlen sor. Egy ilyen sor $0'$-t nem tartalmazhat, és 0^*-ból is legfeljebb egyet, ami kötött oszlopban van. Másrészt a kötött oszlopok száma legfeljebb m, így a sor tartalmaz legalább $n - m$ szabad elemet, azaz léteznek szabad elemek. Mivel nincs szabad 0, és $\mathbf{C}^{(r)}$ elemei nemnegatívak, ezért a szabad elemek rendre pozitívak. Ezek minimumát kivonva minden szabad elemből, az átalakítás utáni mátrix legalább egy szabad 0-t tartalmaz, mégpedig ott, ahol a szabad elemek felveszik a minimumukat. A kapott szabad 0-val vagy a 4. lépéssel, vagy a 3. lépéssel folytatódik az eljárás. Az utóbbi esetben a szabad 0 sorát lekötjük, és a sorában lévő 0^* oszlopát feloldjuk, amiből már következik az 5. lépésre vonatkozó állítás, és az eljárásrész végessége is.

Most megmutatjuk, hogy az 5. lépésre teljesülnek a következők:

(i) az átalakítás során a 0^*, $0'$ elemek nem változnak,

(ii) az átalakítással előállított $\mathbf{C}'^{(r)}$ mátrixra $\mathbf{C}'^{(r)} \geq 0$ és $\mathbf{C}'^{(r)} \sim \mathbf{C}^{(r)}$ teljesül.

Az (i) állítás következik abból, hogy minden egyes 0^* pontosan egyszer

van lekötve vagy a sorában vagy az oszlopában, továbbá minden egyes $0'$ elemnek a sora kötött, az oszlopa pedig szabad.

Az (ii) állításban $\mathbf{C}'^{(r)}$ nemnegatívitása abból adódik, hogy az átalakítás során a szabad elemek pozitív θ minimumát vonjuk ki a szabad elemekből és adjuk hozzá a kétszeresen kötött elemekhez. Végül a $\mathbf{C}'^{(r)} \sim \mathbf{C}^{(r)}$ fennállásának igazolásához vegyük észre, hogy az 5. lépés átalakítása az eredményt illetően megegyezik a következő művelettel: a nem kötött sorok elemeiből rendre vonjunk ki θ-t, a kötött oszlopok elemeihez rendre adjunk hozzá θ-t. Valóban, az utóbbi átalakítás során a kétszeresen kötött elemekhez hozzáadódik θ, az egyszeresen kötött elemek nem változnak, a szabad elemek csökkennek θ-val, azaz a két átalakítás ugyanazt a mátrixot eredményezi. Másrészt a második átalakításról nyilvánvaló, hogy a végrehajtásával előálló $\mathbf{C}'^{(r)}$ mátrix ekvivalens $\mathbf{C}^{(r)}$-rel.

Vizsgáljuk ezek után az eljárás 4. lépését. Mivel minden oszlop legfeljebb egy 0^*-ot, és minden sor legfeljebb egy $0'$-t tartalmaz, ezért a láncképző algoritmus egyértelműen meghatározza a láncot. Jelölje

$$(i_1, \lambda_1), (i_2, \lambda_1), (i_2, \lambda_2), \ldots$$

a láncbeli elemek indexpárjainak a sorozatát a láncképzésnek megfelelő sorrendben. Megmutatjuk, hogy a lánc nem metszi önmagát, azaz a fenti indexpársorozat elemei páronként különbözőek. Ehhez igazoljuk a következőket:

(a) a fenti sorozatban a láncbeli $0'$ elemekhez tartozó indexpárok páronként különbözők,

(b) a láncképzés során minden láncbeli 0^* elemet követ egy $0'$ elem a láncban.

Az (a) állítás igazolásához a mátrix minden egyes $0'$ eleméhez rendeljük hozzá azt az időpontot, amikor az illető 0-t az eljárás során elláttuk ","-vel. Így minden $0'$-höz hozzárendeltünk egy időpontot, és különböző $0'$-khöz különböző időpont tartozik.

$$\begin{array}{ll} 0^* & \cdots \longrightarrow 0'_{\sigma'} \\ \uparrow & \\ | & \\ | & \\ | & \\ \ldots\ 0'_{\sigma} & \end{array}$$

Most tekintsünk a láncban két egymást követő $0'$ elemet. Jelölje az első elemhez rendelt időpontot σ és a második $0'$-höz rendelt időpontot σ'. Vegyük észre, hogy az első $0'$ oszlopában van egy 0^*, így ez az oszlop az iterációs eljárásrész kezdetekor le van kötve. Ezt a kötést a 3. lépés alapján akkor szüntetjük meg, amikor a 0^* sorában ellátunk egy szabad 0-t ","-vel. Mivel minden sor

legfeljebb egy $0'$ elemet tartalmaz, így ez az időpont σ'. De ez azt jelenti, hogy a tekintett 0^* oszlopában lévő bármely 0 elem a σ' időpontban válhat szabad 0-vá, következésképp ","-vel történő megjelölésére csak egy későbbi időpontban kerülhet sor, azaz $\sigma > \sigma'$. Ezzel viszont azt kapjuk, hogy a fenti indexpársorozatban a $0'$-khöz rendelt időpontok egy szigorúan monoton csökkenő sorozatot alkotnak, amiből már következik az állítás.

A (b) állítás igazolásához tekintsük a lánc egy tetszőleges 0^* elemét.

$$\begin{array}{c} 0^* \\ \uparrow \\ | \\ \ldots\ 0' \end{array}$$

A láncképző algoritmus szerint a 0^* oszlopában egy láncbeli $0'$ előzi meg a 0^*-ot. Másrészt a 0^* oszlopa az iterációs eljárás kezdetén le van kötve. Így ebbe az oszlopba csak úgy kerülhetett $0'$, hogy a 0^* sorában egy szabad 0-t elláttunk ","-vel és a 0^* oszlopát feloldottuk. De akkor a láncképző algoritmus szerint ez a $0'$ a lánc következő eleme, amivel az állítást igazoltuk.

Most megmutatjuk, hogy az (i_1, λ_1), (i_2, λ_1), ... sorozatban a láncbeli 0^*-okhoz tartozó indexek különbözők. Valóban, ha két ilyen indexpár megegyezne, akkor (b) alapján az illető 0^*-okat követi egy-egy $0'$, és a láncképzés egyértelműsége miatt ezen $0'$-khöz tartozó indexpárok is megegyeznének, ami ellentmond (a)-nak. Következésképp, a vizsgált indexpárok is páronként különbözők.

Mivel az eljárás során a 0-kat legfeljebb egy jellel láttuk el, ezért a fentiekből már következik, hogy az (i_1, λ_1), (i_2, λ_1), ... indexpárok páronként különbözők, azaz a lánc nem metszi önmagát.

Vizsgáljuk végül a láncon kívüli 0^*-okból és a láncbeli $0'$-kből álló 0-rendszert. Megmutatjuk, hogy ez olyan független 0-rendszer, amelynek elemszáma $m + 1$.

Valóban, a láncon kívüli 0^*-ok független 0-rendszert alkotnak, mert a 0^*-ok független 0-rendszert jelöltek, és egy ilyen rendszer minden részrendszere is független 0-rendszer. Most tekintsük a láncbeli $0'$ elemeket. Mivel minden sor legfeljebb egy $0'$-t tartalmaz, ezért minden sorban legfeljebb egy eleme van a tekintett rendszernek. Megmutatjuk, hogy ez az oszlopokra is teljesül. Tegyük fel, hogy valamely oszlop két láncbeli $0'$ elemet tartalmaz. Akkor ezek közül valamelyik $0'$ kezdő vagy közbülső eleme a láncnak, ezért a vizsgált oszlop tartalmaz egy láncbeli 0^*-ot is. De akkor a láncképző algoritmus szerint a másik $0'$-t is ez a 0^* követné a láncban, és így a lánc metszené önmagát, ami ellentmondás. Következésképp, minden oszlop legfeljebb egy láncbeli $0'$-t tartalmaz, és így a láncbeli $0'$-k független 0-rendszert alkotnak.

Ezek után az új 0-rendszer függetlenségéhez csak azt kell igazolnunk, hogy

(i) láncbeli $0'$ sorában nincs láncon kívüli 0^*,

(ii) láncbeli $0'$ oszlopában nincs láncon kívüli 0^*.

Az (i) állítás igazolásához tegyük fel, hogy valamely láncbeli $0'$ elem sora tartalmaz láncon kívüli 0^*-ot. Ekkor a tekintett $0'$ nem lehet a lánc kezdő eleme, ugyanis csak akkor indíthatunk $0'$-ből láncot, ha a sora nem tartalmaz 0^* elemet. Így a vizsgált $0'$ közbülső vagy befejező eleme a láncnak. De ekkor a láncban van egy elem előtte, mégpedig egy 0^*. A láncképző algoritmus alapján ez a 0^* is a tekintett $0'$ sorában helyezkedne el, és így ez a sor két 0^*-ot tartalmazna, ami ellentmondás. Következésképp, láncbeli $0'$ sora nem tartalmazhat láncon kívüli 0^*-ot.

Hasonlóan látható be az (ii) állítás is. Ehhez tegyük fel, hogy valamely láncbeli $0'$ oszlopa tartalmaz láncon kívüli 0^*-ot. Ekkor a tekintett $0'$ elem nem lehet a lánc befejező eleme, ugyanis a láncképző algoritmus szerint az oszlopban lévő 0^*-gal folytatódna a lánc. Így a tekintett $0'$ a lánc kezdő vagy közbülső eleme. De akkor a láncképző algoritmus szerint ez az oszlop tartalmaz egy láncbeli 0^*-ot is. Ezzel azt kapjuk, hogy az oszlopban két 0^* van, ami ellentmondás. Következésképp, láncbeli $0'$ oszlopa nem tartalmaz láncon kívüli 0^*-ot.

Most felhasználva, hogy mind a láncon kívüli 0^*-ok, mind a láncbeli $0'$-k független 0-rendszert alkotnak, (i) és (ii) alapján egyszerűen adódik, hogy a láncon kívüli 0^*-ok és a láncbeli $0'$-k együttesen egy független 0-rendszert alkotnak.

Hátra van még annak igazolása, hogy az új független 0-rendszer elemeinek a száma $m+1$. Tudjuk, hogy a lánc nem metszi önmagát. Másrészt a lánc $0'$-vel kezdődik és (b) alapján $0'$-vel végződik. Ez pontosan azt jelenti, hogy a láncban szereplő $0'$-k száma eggyel több, mint a láncbeli 0^*-ok száma, amivel adódik az elemszámokra vonatkozó állítás.

Összegezve az eddigieket, megmutattuk, hogy az iterációs eljárásrésszel a $\mathbf{C}^{(r)}$-ből közvetlenül, vagy közbülső mátrixok felhasználásával előállított $\mathbf{C}^{(r+1)}$ mátrixra teljesülnek a következők:

(I) $\mathbf{C}^{(r)} \sim \mathbf{C}^{(r+1)}$,

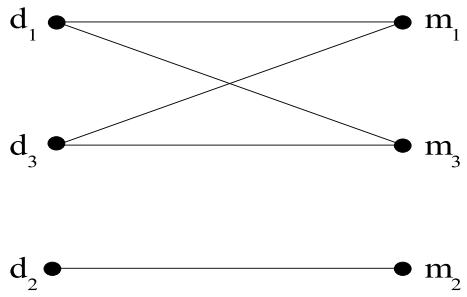
(II) $\mathbf{C}^{(r+1)} \geq 0$,

(III) $\mathbf{C}^{(r+1)}$-ben ki van jelölve egy (m+1)-elemű független 0-rendszer.

Ekkor $\mathbf{C}^{(r+1)}$-re teljesül (2) és (3). Másrészt, mivel minden iteráció során eggyel nő a független 0-rendszer elemeinek a száma, ezért az iterációk száma legfeljebb $n-1$. Ebből már következik az iterációs eljárásrész és egyben az egész eljárás helyessége.

Az alfejezet befejezéseként megoldunk két olyan feladatot, amelyekben a problémak gráfja nem teljes páros gráf. Az első példának létezik optimális megoldása, míg a másikról kiderül majd, hogy nem létezik lehetséges megoldása.

3.2.2. példa. Tekintsük azt a gráfot, melyben a csúcsok halmaza $V = \{d_1, d_2, d_3, m_1, m_2, m_3\}$ és az élek (d_t, m_t), $(t = 1, 2, 3)$, $(d_3, m_1), (d_1, m_3)$. Legyenek az élek súlyai: $c_{tt} = 1$, $(t = 1, 2, 3)$, $c_{13} = c_{31} = 3$. A gráf grafikus reprezentációját adja meg a 3.2.1. ábra.



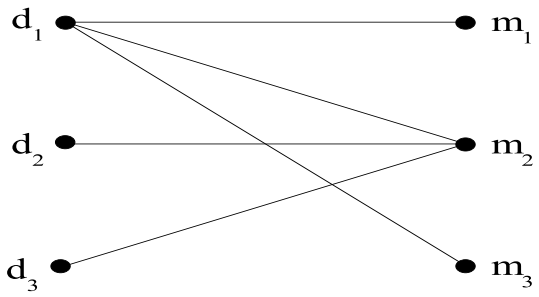
3.2.1. ábra. A 3.2.2. példa gráfjának grafikus reprezentációja.

Most a 3.2.2. segédtétel előkészítésének megfelelően legyen W az élek súlyainak összege plusz 1. Akkor $W = 10$. Ezek után kiegészítve a gráfot a $(d_1, m_2), (d_2, m_1)$ és $(d_3, m_2), (d_2, m_3)$ élekkel, egy teljes páros gráfot kapunk, amelyben az új élekhez rendeljük a W súlyt és a régi élek súlyait tartsuk meg. Akkor az új párosítási feladat (3.2.4) alakban írható fel, amit meg tudunk oldani a magyar módszerrel. A megoldást az alábbi mátrixsorozat szolgáltatja:

$$\begin{pmatrix} 1 & 10 & 3 \\ 10 & 1 & 10 \\ 3 & 10 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 9 & 2 \\ 9 & 0 & 9 \\ 2 & 9 & 0 \end{pmatrix} \quad \begin{pmatrix} 0^* & 9 & 2 \\ 9 & 0^* & 9 \\ 2 & 9 & 0^* \end{pmatrix}$$

Most egy minimális súlyú teljes párosítás az $M = \{(d_1, m_1), (d_2, m_2), (d_3, m_3)\}$ élhalmaz, amelynek a súlya 3. Másrészt $3 < 10 = W$, így a 3.2.2. segédtétel alapján M optimális megoldása (minimális súlyú teljes párosítása) a 3.2.2. példában megadott párosítási feladatnak.

3.2.3. példa. Legyen $\mathcal{G} = (V, E)$, ahol $V = \{d_1, d_2, d_3, m_1, m_2, m_3\}$ és az élek (d_1, m_t), $(t = 1, 2, 3)$, (d_2, m_2), (d_3, m_2). A felsorolt élek mindegyikének legyen a súlya 1. A gráf grafikus reprezentációja a 3.2.2. ábrán látható.



3.2.2. ábra. A 3.2.3. példa gráfjának grafikus reprezentációja.

Legyen W az élek súlyainak az összege plusz 1, azaz $W = 6$. Egészítsük ki $\mathcal{G}$-t a (d_2, m_1), (d_2, m_3) és (d_3, m_1), (d_3, m_3) élekkel. Akkor az új gráf már teljes páros gráf lesz. A felvett 4 új él mindegyikének legyen a súlya W. Ekkor az új párosítási feladat (3.2.4) alakban írható fel, amelynek megoldását a következő mátrixsorozat szolgáltatja.

$$\begin{array}{ccccccc} & & \oplus & + & & \oplus & \\ \begin{pmatrix} 1 & 1 & 1 \\ 6 & 1 & 6 \\ 6 & 1 & 6 \end{pmatrix} & \begin{pmatrix} 0^* & 0 & 0' \\ 5 & 0^* & 5 \\ 5 & 0 & 5 \end{pmatrix} + & \begin{pmatrix} 0^* & 5 & 0' \\ 0 & 0^* & 0' \\ 0 & 0 & 0' \end{pmatrix} + & \begin{pmatrix} 0^* & 0 & 0 \\ 0 & 0^* & 0 \\ 0 & 0 & 0^* \end{pmatrix} \end{array}$$

Azt kapjuk, hogy az $M = \{(d_t, m_t) : t = 1, 2, 3\}$ egy minimális súlyú teljes párosítás az új feladatban. Másrészt M súlya 8, ami nagyobb, mint W, így a 3.2.2. segédtételből következik, hogy nincs optimális megoldása, és így lehetséges megoldása sem a tekintett 3.2.3. példában megadott feladatnak.

Az alfejezet lezárásaként megemlítjük, hogy 1957-ben J. Munkres [144] igazolta, hogy az ismertetett eljárás műveletigénye $O(n^3)$. Később $A(\mathbf{C})$ megoldására kidolgozásra kerültek hatékonyabb algoritmusok is, ld. például a [62] és [65] dolgozatokat.

3.3. Minimális súlyú teljes párosítási feladat tetszőleges gráfban

Ebben az alfejezetben az előzőekben tárgyalt párosítási problémának azt az általánosítását fogjuk vizsgálni, amelyben nem tételezzük fel, hogy a tekintett gráf páros gráf. A probléma megfogalmazásához legyen adott egy $\mathcal{G} = (V, E)$ gráf és legyen $\mathcal{G}$ minden $(u, v) \in E$ éléhez hozzárendelve egy w_{uv} súly. Feladat $\mathcal{G}$ egy minimális súlyú teljes párosításának meghatározása, azaz keressük a

$$\min\{\textstyle\sum_{(u,v)\in M} w_{uv} : M \ \mathcal{G} \text{ teljes párosítása }\} \tag{3.3.1}$$

feladat egy optimális megoldását. Nyilvánvaló, hogy $\mathcal{G}$-nek csak akkor létezhet teljes párosítása, ha csúcspontjainak száma páros. Ezért a továbbiakban minden hivatkozás nélkül feltételezzük, hogy a (3.3.1) problémában $|V| = 2n$, ahol n pozitív egész. Vegyük észre, hogy a (3.3.1) probléma esetében is az általánosság megszorítása nélkül feltehető, hogy a w_{uv} súlyok nemnegatívak. Ez ugyanúgy igazolható, mint ahogy azt a 3.2.1. segédtételnél tettük. Így a továbbiakban azt is feltételezzük, hogy a (3.3.1) feladatban a w_{uv} súlyok mindegyike nemnegatív.

A (3.3.1) feladat vizsgálata előtt tekintsük annak egy gyakorlati alkalmazását.

Raktárfelszámolási feladat

Adott egy raktár, amely $2n$ páronként különböző tartalmú konténert tartalmaz. A szállítójárművek olyanok, hogy pontosan két konténer szállítására alkalmasak, és csak bizonyos párosokat képesek elszállítani. Minden szállítható konténerpárra ismert az elszállítás költsége, amely függhet az illető konténerektől, a fogadó állomástól, a fogadóhelynek a raktártól való távolságától és számos további dologtól. Feladat a raktár egy olyan kiürítése, amely minimális szállítási költséggel jár.

Tekintsük azt a gráfot, amelynek csúcspontjai a különböző tartalmú konténerek, és az i, j konténereket akkor köti össze él, ha együtt szállíthatók. Az i, j csúcsokat összekötő él súlya legyen az i, j konténerpár szállítási költsége. Ha az i, j csúcsokat több él is összeköti, akkor ezek közül csak egy minimális súlyút tartsunk meg. Akkor a tekintett hálózat egy minimális súlyú teljes párosítása a kiürítési feladat egy optimális megoldása.

A (3.3.1) probléma megoldása visszavezethető egy másfajta párosítási probléma megoldására. Ebből a célból megkonstruálunk a (3.3.1) problémához egy további párosítási feladatot.

Legyen $W = 1 + \sum_{(u,v)\in E} w_{uv}$ és minden $(u,v) \in E$ élre $w'_{uv} = W - w_{uv}$. Tekintsük most az alábbi párosítási feladatot:

(3.3.2) $\qquad \max\{\sum_{(u,v)\in M} w'_{uv} : \ M\ \mathcal{G}$ párosítása $\}$

azaz $\mathcal{G}$ egy maximális súlyú párosítását keressük a w'_{uv} súlyok mellett.

A tekintett két feladat között szoros kapcsolat van, amint azt a következő állítás is mutatja.

3.3.1. segédtétel. *A (3.3.1) feladatnak akkor és csak akkor létezik optimális megoldása, ha (3.3.2) optimuma nem kisebb, mint* $(n-1)W+1$ *és ez esetben (3.3.2) optimális megoldása egyben (3.3.1) feladatnak is optimális megoldása.*

Bizonyítás. Az állítás korrekt, mivel a (3.3.2) feladatnak mindig létezik optimális megoldása. Jelölje rendre w_1 és w_2 a (3.3.1) és (3.3.2) feladatok célfüggvényeit.

Elsőként tegyük fel, hogy a (3.3.1) feladatnak létezik egy M optimális teljes párosítása. Akkor ez lehetséges megoldása a (3.3.2) feladatnak és

$$w_2(M) = \sum_{(u,v)\in M} w'_{uv} = \sum_{(u,v)\in M} (W - w_{uv}) = nW - \sum_{(u,v)\in M} w_{uv} \geq (n-1)W+1$$

ugyanis M n számú élet tartalmaz, továbbá

$\sum_{(u,v)\in M} w_{uv} \leq \sum_{(u,v)\in E} w_{uv} = W - 1.$

A fentiek alapján M olyan lehetséges megoldása a (3.3.2) feladatnak, amelyre $w_2(M) \geq (n-1)W + 1$ teljesül. Ebből már közvetlenül adódik, hogy (3.3.2) optimuma nem kisebb, mint $(n-1)W + 1$.

Most tegyük fel, hogy (3.3.2) optimuma nem kisebb, mint $(n-1)W+1$. Legyen az M párosítás (3.3.2) egy optimális megoldása. Akkor $w_2(M) \geq (n-1)W + 1$. Most vegyük észre, hogy ebből az egyenlőtlenségből $|M| = n$ következik. Valóban $|M| = k$, $k < n$ esetén,

$$w_2(M) = kW - \sum_{(u,v)\in M} w_{uv} \leq kW < (n-1)W + 1,$$

ami ellentmondás. Következésképpen M n számú élet tartalmazó párosítása $\mathcal{G}$-nek, de akkor M egy teljes párosítása $\mathcal{G}$-nek. Ezzel azt kapjuk, hogy a

vizsgált esetben létezik (3.3.1)-nek lehetséges megoldása, de akkor létezik optimális megoldása is.

Ezek után azt kell még igazolnunk, hogy a vizsgált esetben (3.3.2) minden optimális megoldása optimális megoldása (3.3.1)-nek is. Ehhez legyen M (3.3.2)-nek egy olyan optimális megoldása, amelyre $w_2(M) \geq (n-1)W+1$ teljesül. Az előzőek alapján tudjuk, hogy ekkor M teljes párosítása $\mathcal{G}$-nek, azaz lehetséges megoldása (3.3.1)-nek. Allítjuk, hogy M optimális megoldása (3.3.1)-nek. Ezt indirekt igazoljuk. Tegyük fel, hogy M nem optimális megoldása (3.3.1)-nek. Akkor van olyan M' teljes párosítása $\mathcal{G}$-nek, hogy $0 \leq w_1(M') < w_1(M)$. Mivel M' párosítása $\mathcal{G}$-nek, ezért lehetséges megoldása (3.3.2)-nek is. De akkor

$$w_2(M) = nW - \sum_{(u,v)\in M} w_{uv} = nW - w_1(M) < \\ nW - w_1(M') = nW - \sum_{(u,v)\in M'} w_{uv} = w_2(M'),$$

azaz $w_2(M) < w_2(M')$, ami ellentmond annak, hogy M a (3.3.2) feladat optimális megoldása. Következésképpen M optimális megoldása (3.3.1)-nek, amivel a 3.3.1. segédtételt igazoltuk.

Nem használjuk, de megemlítjük, hogy erősebb kapcsolat van a tekintett két feladattípus között, mint amit a 3.3.1. segédtételben megfogalmaztunk. Nevezetesen a (3.3.2) feladat megoldása is visszavezethető egy alkalmas (3.3.1) típusú feladat megoldására. A visszavezetés megtalálható például a [112] könyvben.

Vegyük észre, hogy a (3.3.1) feladathoz rendelt (3.3.2) feladatban szereplő súlyok mindegyike pozitív, ha a (3.3.1) feladat súlyai nemnegatívak. Mivel ez utóbbi tulajdonság az általánosság megszorítása nélkül feltehető, ezért a továbbbiakban olyan (3.3.2) típusú feladatok megoldásával foglalkozunk, amelyekben a súlyok mindegyike pozitív.

Nyilvánvaló módon a (3.3.2) feladat megoldása során közömbös, hogy a gráf csúcsainak száma páros vagy páratlan. Ezért a továbbiakban feltételezzük, hogy a tekintett $\mathcal{G} = (V, E)$ gráfra $V = \{1, \ldots, n\}$ és az $(i, j) \in E$ élhez rendelt pozitív súlyt w_{ij} jelöli.

A (3.3.2) feladat megoldása visszavezethető egy alkalmas lineáris programozási feladat megoldására a következő módon. Ehhez elsőként indexezzük az éleket, azaz legyen $E = \{e_1, \ldots, e_m\}$. Ekkor E tetszőleges A részhalmazát leírhatjuk egy m-dimenziós bináris $\bar{\mathbf{x}}$ vektorral, amit a következőképpen definiálunk. Minden $1 \leq j \leq m$ egészre legyen

$$\bar{x}_j = \begin{cases} 1, & \text{ha } e_j \in A, \\ 0 & \text{különben.} \end{cases}$$

Az $\bar{\mathbf{x}}$ vektort az A halmaz *karakterisztikus vektorának* nevezzük. Egyszerűen belátható, hogyha E minden részhalmazához hozzárendeljük annak karakterisztikus vektorát, akkor E részhalmazainak egy kölcsönösen egyértelmű leképezését kapjuk az m-dimenziós bináris vektorok halmazára. Tekintsük most a V halmaz összes lehetséges 1-nél nagyobb páratlan elemszámú részhalmazát. Jelölje ezen részhalmazokat $U_1, \ldots, U_l$ és elemszámaikat rendre $2n_1 + 1, \ldots, 2n_l + 1$. Minden $j \in \{1, \ldots, l\}$ egészre jelölje S_j azon E-beli élek halmazát, amelyeknek mindkét végpontja eleme U_j-nek. Végül minden $e_t \in E$ élre legyen $w_t = w_{uv}$, ha $e_t = (u, v)$. Most tekintsük a következő lineáris programozási feladatot:

$$\begin{array}{c} \sum_{\{t:\, i \text{ illeszkedik } e_t\text{-re}\}} x_t \leq 1,\ (i = 1, \ldots, n) \\ \sum_{\{t:\, e_t \in S_j\}} x_t \leq n_j,\ (j = 1, \ldots, l) \\ x_t \in \{0, 1\},\ (t = 1, \ldots, m) \\ \hline \sum_{\{t:\, e_t \in E\}} w_t x_t \to \max \end{array} \tag{3.3.3}$$

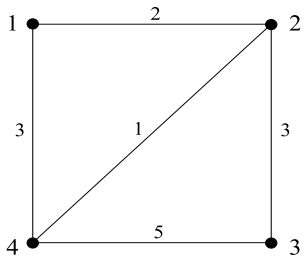
A fenti lineáris programozási feladatban $\mathcal{G}$ párosításait karakterisztikus vektoraikkal írjuk le. Az első feltételcsoport azt biztosítja, hogy minden $i \in V$ csúcsra, az i csúcs legfeljebb egy M-beli élnek végpontja. A második feltételcsoport azt garantálja, hogy minden U_j $2n_j + 1$ elemszámú csúcshalmazra legfeljebb n_j számú olyan M-beli él van, amelyeknek végpontjai U_j-beli csúcsok. Ezek után egyszerűen belátható, hogy (3.3.3) lehetséges megoldásai pontosan a $\mathcal{G}$ párosításai. Mivel egy M párosítás súlya megegyezik a karakterisztikus vektorán felvett célfügvényértékkel, ezért a (3.3.3) feladat optimális megoldásához tartozó párosítás optimális megoldása a (3.3.2) feladatnak és fordítva.

A (3.3.3) feladat megoldását nagyban megnehezíti annak nagy mérete, továbbá az, hogy egészértékű (speciálisan bináris) lineáris programozási feladat. Szerencsére az utóbbi problémát nem kell kezelni, mivel (3.3.3) speciális struktúrája biztosítja, hogy lineáris programozási relaxációjának (az $x_t \in \{0, 1\},\ (t = 1, \ldots, m)$ feltételeket kicseréljük a $0 \leq x_t,\ (t = 1, \ldots, m)$ feltételekre) optimális megoldása egyben a (3.3.3) feladatnak is optimális megoldása. Következésképpen, (3.3.2) megoldásához elegendő a következő lineáris programozási feladatot megoldani.

$$\begin{array}{c} \sum_{\{t:\, i \text{ illeszkedik } e_t\text{-re}\}} x_t \leq 1,\ (i = 1, \ldots, n) \\ \sum_{\{t:\, e_t \in S_j\}} x_t \leq n_j,\ (j = 1, \ldots, l) \\ 0 \leq x_t,\ (t = 1, \ldots, m) \\ \hline \sum_{\{t:\, e_t \in E\}} w_t x_t \to \max \end{array} \tag{3.3.4}$$

Az eddigiek illusztrálására tekintsünk egy kicsi és triviális párosítási feladatot.

3.3.1. példa. Legyen $\mathcal{G} = (\{1, \ldots, 4\}, E)$, ahol $E = \{(1,2), (1,4), (2,3), (2,4), (3,4)\}$, az élek az adott felsorolás szerint vannak sorszámozva, és $w_1 = 2, w_2 = 3, w_3 = 3, w_4 = 1, w_5 = 5$. A tekintett hálózat grafikus reprezentációját mutatja a 3.3.1. ábra.



3.3.1. ábra. A 3.3.1. példa hálózatának grafikus reprezentációja.

A feladat optimális megoldása nyilvánvalóan az $M = \{(1,2), (3,4)\}$ párosítás. Legyen $U_1 = \{1,2,3\}$, $U_2 = \{1,2,4\}$, $U_3 = \{1,3,4\}$, $U_4 = \{2,3,4\}$. Akkor a megfelelő lineáris programozási feladat szimplex táblázata a következő:

	x_1	x_2	x_3	x_4	x_5		
y_1	1	1	0	0	0	1	
y_2	1	0	1	1	0	1	
y_3	0	0	1	0	1	1	
y_4	0	1	0	1	1	1	
z_1	1	0	1	0	0	1	(U_1)
z_2	1	1	0	1	0	1	(U_2)
z_3	0	1	0	0	1	1	(U_3)
z_4	0	0	1	1	1	1	(U_4)
	2	3	3	1	5		

Szimplex algoritmussal megoldva a lineáris programozási feladatot, azt kapjuk, hogy az optimális megoldás, az $\bar{\mathbf{x}} = (1,0,0,0,1)$ vektor, az optimum értéke pedig 7.

A továbbiakban az [54] könyvre alapozva egy, a (3.3.2) feladat megoldására szolgáló eljárást körvonalazunk. Ehhez szükségünk lesz néhány, korábbról már ismert fogalom és összefüggés felidézésére.

A következő lineáris programozási feladatot szokásos *primál feladatnak* nevezni.

$$\frac{\mathbf{A}^{n\times m}\mathbf{x} \le \mathbf{b},\ \mathbf{x} \ge 0}{\mathbf{cx} = z(\mathbf{x}) \to \max}$$

A fenti feladat *duális feladatán* a következő feladatot értjük.

$$\frac{\mathbf{yA} \ge \mathbf{c},\ \mathbf{y} \ge 0}{\mathbf{yb} = w(\mathbf{y}) \to \min}$$

Most legyen $\bar{\mathbf{x}}$ a primál feladat és $\bar{\mathbf{y}}$ a duál feladat egy lehetséges megoldása. Azt mondjuk, hogy az $\bar{\mathbf{x}}$ és $\bar{\mathbf{y}}$ megoldáspár teljesíti a *komplementaritási feltételeket*, ha teljesülnek a következők:

$$\bar{y}_i(b_i - \sum_{t=1}^{m} a_{it}\bar{x}_t) = 0,\ (i = 1, \ldots, n)$$

$$\bar{x}_j(\sum_{t=1}^{n} a_{tj}\bar{y}_t - c_j) = 0,\ (j = 1, \ldots, m)$$

A primál és duál feladatpár valamint a komplementaritási feltételek közötti kapcsolatra a következő állítás érvényes.

3.3.1. tétel. (Komplementaritási tétel.) *Legyen $\bar{\mathbf{x}}$ a primál feladat és $\bar{\mathbf{y}}$ a duál feladat lehetséges megoldása. Az $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ akkor és csak akkor optimális megoldások, ha kielégítik a komplementaritási feltételeket.*

A tételt nem igazoljuk, annak bizonyítása megtalálható például a [7], [54], [168] munkákban.

Vegyük észre, hogy a (3.3.4) feladat egy primál feladat. Egyszerűen ellenőrizhető, hogy a (3.3.4) feladat duális feladata a következő lineáris programozási feladat:

$$\begin{array}{l} y_i + y_j + \sum_{\{t:\,(i,j)\in S_t\}} z_t \ge w_{ij},\ ((i,j) \in E) \\ y_i \ge 0,\ (i = 1, \ldots, n)\ \ z_t \ge 0,\ (t = 1, \ldots, l) \\ \hline \sum_{i=1}^{n} y_i + \sum_{t=1}^{l} n_t z_t = w \to \min \end{array} \qquad (3.3.5)$$

A (3.3.5) duális feladat illusztrálására megkonstruáljuk a 3.3.1. példához tartozó primál feladat duálisát. A duális feladat szimplex táblázatát adjuk meg, ahol az egyenlőtlenségek -1-szerese szerepel, és az egyes feltételek mellett megadjuk, hogy az illető feltétel melyik élhez tartozik.

	y_1	y_2	y_3	y_4	z_1	z_2	z_3	z_4		
x_1	-1	-1	0	0	-1	-1	0	0	-2	$(1,2)$
x_2	-1	0	0	-1	0	-1	-1	0	-3	$(1,4)$
x_3	0	-1	-1	0	-1	0	0	-1	-3	$(2,3)$
x_4	0	-1	0	-1	0	-1	0	-1	-1	$(2,4)$
x_5	0	0	-1	-1	0	0	-1	-1	-5	$(3,4)$
	1	1	1	1	1	1	1	1		

Mivel az U_1, U_2, U_3 és U_4 csúcshalmazok mindegyike háromelemű, ezért a z_t változók célfüggvényegyütthatói a fenti feladatban $n_1 = n_2 = n_3 = n_4 = 1$.

Most felírjuk a komplementaritási feltételeket a (3.3.4) és (3.3.5) feladatpárra kicsit módosított alakban. A feltételeket három csoportban adjuk meg.

(i) $(\forall e_t = (i,j) \in E\text{-re})\ (x_t > 0 \Longrightarrow (y_i + y_j + \sum_{\{t:\ (i,j)\in S_t\}} z_t = w_{i,j}))$.

Mivel minden élre van egy ilyen feltétel, ezért ezen csoportban a feltételek száma m.

(ii) $(\forall y_i)\ (y_i > 0 \Longrightarrow \sum_{\{t:\ i \text{ illeszkedik } e_t-\text{re}\}} x_t = 1)$.

Mivel ebben a feltételcsoportban minden i csúcsra képezünk egy feltételt, ezért az (ii) típusú feltételek száma megegyezik a csúcspontok n számával.

(iii) $(\forall z_j)\ (z_j > 0 \Longrightarrow \sum_{\{t:\ e_t\in S_j\}} x_t = n_j.)$

Az (iii) feltételcsoportban minden páratlan számú csúcsból álló U_j halmazra képezünk egy feltételt, ezért az ilyen feltételek száma l.

A 3.3.1. példához tartozó primál-duál feladatpár komplementaritási feltételei a következők:

(i) $x_1 > 0 \Longrightarrow\ (y_1 + y_2 + z_1 + z_2 = 2)$
$x_2 > 0 \Longrightarrow\ (y_1 + y_4 + z_2 + z_3 = 3)$
$x_3 > 0 \Longrightarrow\ (y_2 + y_3 + z_1 + z_4 = 3)$
$x_4 > 0 \Longrightarrow\ (y_2 + y_4 + z_2 + z_4 = 1)$
$x_5 > 0 \Longrightarrow\ (y_3 + y_4 + z_3 + z_4 = 5)$

(ii) $y_1 > 0 \Longrightarrow (x_1 + x_2 = 1)$
$y_2 > 0 \Longrightarrow (x_1 + x_3 + x_4 = 1)$
$y_3 > 0 \Longrightarrow (x_3 + x_5 = 1)$
$y_4 > 0 \Longrightarrow (x_2 + x_4 + x_5 = 1)$

(iii) $z_1 > 0 \Longrightarrow (x_1 + x_3 = 1)$
$z_2 > 0 \Longrightarrow (x_1 + x_2 + x_4 = 1)$
$z_3 > 0 \Longrightarrow (x_2 + x_5 = 1)$
$z_4 > 0 \Longrightarrow (x_3 + x_4 + x_5 = 1)$

A (3.3.2) feladat megoldására szolgáló eljárás alapgondolata a következő. Az eljárás során meghatározunk egy

$$(\mathbf{x}^{(0)}, (\mathbf{y}^{(0)}, \mathbf{z}^{(0)})), (\mathbf{x}^{(1)}, (\mathbf{y}^{(1)}, \mathbf{z}^{(1)})), \ldots, (\mathbf{x}^{(k)}, (\mathbf{y}^{(k)}, \mathbf{z}^{(k)}))$$

sorozatot úgy, hogy

(1) $\mathbf{x}^{(t)}$ lehetséges megoldása a (3.3.4) feladatnak, $(t = 0, \ldots, k)$,

(2) $(\mathbf{y}^{(t)}, \mathbf{z}^{(t)})$ lehetséges megoldása a (3.3.5) feladatnak, $(t = 0, \ldots, k)$,

(3) az $\mathbf{x}^{(t)}$ és $(\mathbf{y}^{(t)}, \mathbf{z}^{(t)})$ megoldáspár kielégíti az (i) és (iii) feltételeket, $(t = 1, \ldots, k)$,

(4) az $\mathbf{x}^{(k)}$ és $(\mathbf{y}^{(k)}, \mathbf{z}^{(k)})$ megoldáspár kielégít az (ii) feltételcsoport összes feltételét.

Ha meg tudunk határozni egy ilyen megoldáspár sorozatot, akkor $\mathbf{x}^{(k)}$ (3.3.4) lehetséges megoldása, $(\mathbf{y}^{(k)}, \mathbf{z}^{(k)})$ (3.3.5) lehetséges megoldása, és ezen két megoldás kielégíti az összes komplementaritási feltételt. Ekkor a 3.3.1. tétel alapján $\mathbf{x}^{(k)}$ optimális megoldása a (3.3.4) feladatnak. Ezek után véve az $\mathbf{x}^{(k)}$ karakterisztikus vektor által meghatározott $M(\subseteq E)$ élhalmazt, M optimális megoldása lesz a (3.3.2) feladatnak.

Ezek után már csak az a kérdés, hogy miként tudunk ilyen sorozatot előállítani. Erre adott meg egy eljárást 1970-ben J. Edmonds és E. Johnson [48]. Az általuk kidolgozott eljárás alkalmazza az alternáló fa eljárást és az abból kapott eredmények alapján állítja elő az egymást követő megoldáspárokat.

Fontosnak tartjuk megjegyezni, hogy az $\mathbf{x}^{(t)}$, $\mathbf{y}^{(t)}$, $\mathbf{z}^{(t)}$ vektorokat így elkülönítve sem az eljárásban, sem a további példákban nem határozzuk

meg. Ezek a vektorok az eljárás során előállnak az $\mathbf{x}$, $\mathbf{y}$, és $\mathbf{z}$ vektorváltozók aktuális értékeként. A példákban megadjuk ezen értékeket táblázatokban.

Edmonds és Johnson eljárása ([48])

Előkészítő rész

Legyen $\mathbf{x} = \mathbf{0}$ (azaz induljunk az $M_0 = \emptyset$ párosítással). Legyen $c = \max\{w_{uv} : (u,v) \in E\}$, $y_i = c/2$ $(i = 1, \ldots, n)$, $z_t = 0$, $(t = 1, \ldots, l)$, $r = 0$, $\mathcal{G}_r = \mathcal{G}$.

Iterációs rész (r. iteráció)

- *1. lépés.* (Ellenőrzés.) Válasszunk $\mathcal{G}_r$-ben egy olyan v nem mesterséges és M_r-re nézve szabad csúcsot , amelyre $y_v > 0$. Ha nincs ilyen csúcs $\mathcal{G}_r$-ben, akkor a 7. lépés következik. Ellenkező esetben legyen $s = v$, adjuk az s csúcsnak az *outer* címkét, és legyen s az aktuális alternáló fa gyökere, majd folytassuk az eljárást a 2. lépéssel.

- *2. lépés.* Legyen E^* mindazon $\mathcal{G}_0$-beli (i,j) élek halmaza, amelyekre $y_i + y_j + \sum_{\{t:\,(i,j)\in S_t\}} z_t = w_{ij}$ teljesül. (Ha egy $e_t \in E^*$ élet bevonunk egy párosításba, akkor az (i) feltételcsoport megfelelő feltétele teljesül.) Térjünk rá a 3. lépésre.

- *3. lépés.* Folytassuk az alternáló fa eljárást úgy, hogy a faépítés során az E^*-beli éleket és az M_r párosítást használjuk.

  Ha az alternáló fa eljárás M_r-re nézve egy $\mathcal{P}$ javító utat eredményez, akkor a 4. lépés következik.

  Ha az alternáló fa eljárás magyar fát eredményez, akkor az eljárás az 5. lépéssel folytatódik.

  Ha az alternáló fa eljárás páratlan élszámú kör megtalálásával végződik, akkor a 6. lépés következik.

- *4. lépés.* (Javító út.) Javítsuk az M_r párosítást a megtalált $\mathcal{P}$ javító úttal. A javított párosítást jelölje M_{r+1}. Ha az M_{r+1}-beli élek csúcspontjai között nincs mesterséges csúcs, akkor legyen $\mathbf{x}$ az M_{r+1} párosítás karakterisztikus vektora, $\mathcal{G}_{r+1} = \mathcal{G}_r$. Növeljük r értékét 1-gyel és térjünk rá a következő iterációs lépésre.

- *5. lépés.* (Magyar fa.) Legyen

  $I_1 = \{(i,j) : (i,j) \in E \ \& \ i$ egy *outer* címkéjű csúcsa az alternáló fának, j pedig címkétlen $\}$,

  $$\delta_1 = \min_{(i,j)\in I_1} \{y_i + y_j - w_{ij}\},$$

  $I_2 = \{(i,j) : (i,j) \in E \ \& \ i$ és j *outer* címkéjű csúcsai az alternáló fának, és egyikőjük sincs zsugorított körben $\}$,

  $$\delta_2 = 1/2 \min_{(i,j)\in I_2} \{y_i + y_j - w_{ij}\},$$

  $I_3 = \{k : 1 \le k \le l \ \&$ az U_k csúcsait tartalmazó kör zsugorítva lett egy a^* mesterséges csúccsá és a^* címkéje *inner* $\}$,

  $$\delta_3 = 1/2 \min_{k\in I_3} z_k,$$

  $I_4 = \{i : i \in V \ \& \ i$ címkéje *outer* $\}$,

  $$\delta_4 = \min_{i\in I_4} y_i$$

  Ha valamelyik minimum a fentiekből nem létezik, akkor legyen a megfelelő δ_t értéke ∞. Végül legyen

  $$\delta = \min\{\delta_1, \delta_2, \delta_3, \delta_4\}.$$

  Most változtassuk meg az $\mathbf{y}$ és $\mathbf{z}$ vektorokat a következők szerint:

  (a) csökkentsük az y_t értékét δ-val, ha a t csúcs címkéje *outer*,

  (b) növeljük y_t értékét δ-val, ha a t csúcs címkéje *inner*,

  (c) növeljük a z_t értékét 2δ-val, ha $\mathcal{G}_r$-ben van olyan a^* mesterséges csúcs, amely az U_t csúcshalmaz csúcsait tartalmazó kör zsugorításával keletkezett és az alternáló fa eljárásban *outer* címkét kapott.

  (d) csökkentsük z_t értékét 2δ-val, ha $\mathcal{G}_r$-ben van olyan a^* mesterséges csúcs, amely az U_t csúcshalmaz csúcsait tartalmazó kör zsugorításával keletkezett és az alternáló fa eljárásban *inner* címkét kapott.

  Ha $\delta = \delta_1$, akkor az az (i,j) él, amelyik meghatározta a δ_1 értékét bekerül E^*-ba. Így majd ezt az élet fel lehet használni az alternáló fa

eljárásban. (Lehet több ilyen él is.) Ebben az esetben menjünk a 2. lépésre, (ahol folytatjuk az aktuális alternáló fa építését).

Ha $\delta = \delta_2$, akkor az az (i, j) él, amelyik meghatározta a δ_2 értéket bekerül E^*-ba, és így fel lehet használni az alternáló fa eljárásban, ahol majd egy páratlan számú élből álló kört eredményez. Ebben az esetben is folytassuk az eljárást a 2. lépéssel, ahol folytatódik az alternáló fa építése.

Ha $\delta = \delta_3$, akkor valamely z_t változó 0-vá válik. Ez akkor fordul elő, ha az iterációs lépés során, valamikor korábban egy, az U_t csúcshalmaz csúcsait tartalmazó kört egy a^* mesterséges változóvá zsugorítottunk. Vegyük $\mathcal{G}_r$ a^*-ra vonatkozó rekonstrukcióját, amit jelöljön $\mathcal{G}_{r+1}$, továbbá jelölje az M_r által generált párosítást $\mathcal{G}_{r+1}$-ben M_{r+1}. Ezek után növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

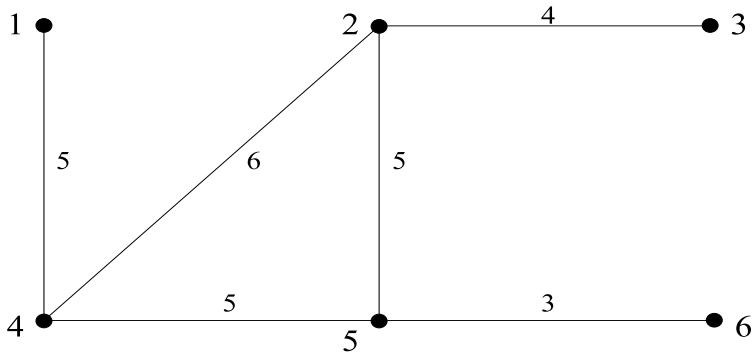
Ha $\delta = \delta_4$, akkor van olyan *outer* címkéjű i csúcs, hogy y_i 0-vá válik. Ekkor az aktuális alternáló fa gyökérből az i csúcsba vezet egy olyan alternáló út, amelyben megegyezik az M_r-beli és nem M_r-beli élek száma. M_r-ből hagyjuk el a tekintett útban szereplő éleket és a maradék élhalmazhoz vegyük hozzá a javító útban szereplő nem M_r-beli éleket. Az új élhalmaz legyen M_{r+1}, továbbá legyen $\mathcal{G}_{r+1} = \mathcal{G}_r$. Növeljük r értékét 1-gyel, majd térjünk rá a következő iterációs lépésre.

- *6. lépés.* (Páratlan élszámú kör.) Növeljük r értékét 1-gyel. Legyen $\mathcal{C}_r$ az alternáló fa eljárással talált kör. Zsugorítsuk a $\mathcal{G}_{r-1}$ gráfot a $\mathcal{C}_r$ körrel, legyen a $\mathcal{C}_r$-nek megfelelő mesterséges változó a^*. Jelölje a zsugorított gráfot $\mathcal{G}_r$ és legyen M_r az M_{r-1} párosítás által indukált párosítás $\mathcal{G}_r$-ben. Vegyük az aktuális alternáló fa s gyökerének $\mathcal{G}_r$-beli képét, ami s, ha az s csúcs nem csúcsa a $\mathcal{C}_r$ körnek, és a^* különben. Jelölje ezt a csúcsot s. Adjuk az s csúcsnak az *outer* címkét és legyen s az aktuális alternáló fa gyökere, majd folytassuk az eljárást a 2. lépéssel.

- *7. lépés.* Ha nincs mesterséges csúcs $\mathcal{G}_r$-ben, akkor vége az eljárásnak, M_r maximális súlyú párosítása $\mathcal{G}$-nek. Ellenkező esetben vegyük $\mathcal{G}_r$-ben azt az a^* mesterséges csúcsot, amely az eljárás során utolsóként lett zsugorítással kialakítva. Jelölje $\mathcal{G}_{r-1}$ $\mathcal{G}_r$ a^*-ra vonatkozó rekonstrukcióját. Legyen M_{r-1} az M_r által generált párosítás $\mathcal{G}_{r-1}$-ben. Csökkentsük r értékét 1-gyel és ismételjük a 7. lépést.

Az eljárás demonstrálására tekintsük a következő feladatot.

3.3.2. példa. ([54]) Legyen $\mathcal{G} = (\{1, \ldots, 6\}, E)$, ahol $E = \{(1,4), (2,3), (2,4), (2,5), (4,5), (5,6)\}$, az élek az adott felsorolás szerint vannak sorszámozva, és $w_1 = 5, w_2 = 4, w_3 = 6, w_4 = 5, w_5 = 5, w_6 = 3$. A tekintett hálózat grafikus reprezentációját mutatja a 3.3.2. ábra.



3.3.2. ábra. A 3.3.2. példa hálózatának grafikus reprezentációja.

A 3.3.2. feladatot leíró primál feladat szimplex táblázata a következő:

	x_1	x_2	x_3	x_4	x_5	x_6	
y_1	1	0	0	0	0	0	1
y_2	0	1	1	1	0	0	1
y_3	0	1	0	0	0	0	1
y_4	1	0	1	0	1	0	1
y_5	0	0	0	1	1	1	1
y_6	0	0	0	0	0	1	1
z_1							n_1
$\vdots$							$\vdots$
z_{14}	0	0	1	1	1	0	n_{14}
$\vdots$							$\vdots$
z_{26}							n_{26}
	5	4	6	5	5	3	

Összesen 26 darab z változónk van. Egy kivételével ezek mindegyike 0 értékű lesz az eljárás során. Az egy kivétel a z_{14} változó, amely az $U_{14} = \{2, 4, 5\}$ csúcshalmazhoz tartozik.

Mivel az eljárás során az $\mathbf{x}$ vektorváltozó aktuális értéke mindig egy párosítás karakterisztikus vektora, ezért ez a vektor lehetséges megoldása a primál feladatnak. Tehát nem kell a primál feladat feltételeinek tejesülését vizsgálni.

A duális feladat feltételei a következők:

$$y_i + y_j + \sum_{\{t:\,(i,j)\in S_t\}} z_t \geq w_{ij},\ ((i,j) \in E)$$
$$y_i \geq 0,\ (i = 1, \ldots, n)\quad z_t \geq 0,\ (t = 1, \ldots, l)$$

A $\mathbf{z} = 0$ és $y_j = c/2$, $(j = 1, \ldots, n)$ értékadással a $\mathbf{z}$ és $\mathbf{y}$ vektorváltozók kezdő értéke a duális feladat egy lehetséges megoldása. Az eljárás során $\mathbf{z}$ és $\mathbf{y}$ értékét úgy változtatjuk, hogy az új vektorpár is lehetséges megoldása legyen a duális feladatnak. Következésképpen, nem kell a duál feladat feltételeinek tejesülését vizsgálni.

A komplementaritási feltételek az alábbiak.

(i) $(\forall e_t = (i,j) \in E\text{-re})\ (x_t > 0 \Longrightarrow y_i + y_j + \sum_{\{t:\,(i,j)\in S_t\}} z_t = w_{i,j})$.
(ii) $(\forall y_i)\ (y_i > 0 \Longrightarrow \sum_{\{t:\, i \text{ illeszkedik } e_t\text{-re}\}} x_t = 1)$.
(iii) $(\forall z_j)\ (z_j > 0 \Longrightarrow \sum_{\{t:\, e_t \in S_j\}} x_t = n_j.)$

Vegyük észre, hogy az eljárásban mindig az E^* halmazból választunk élet, így ha egy x_t-t 1-nek választunk, akkor e_t szükségképpen E^*-beli él, de akkor teljesíti az (i) csoport megfelelő feltételét. Következésképpen az (i) alatti feltételeket kielégítik az eljárással előállított vektorok.

Mivel az eljárás során csak egyetlen z változónak, a z_{14}-nek lesz 0-tól különböző értéke, ezért az (iii) csoport minden feltétele teljesül egyetlen kivétellel, a z_{14}-re vonatkozó feltétellel.

A fentiekből az következik, hogy a tekintett példában az eljárással előállított vektorokra csak az alábbi feltételek teljesülését kell ellenőrizni:

$$z_{14} > 0 \Longrightarrow x_3 + x_4 + x_5 = 1,$$
$$y_1 > 0 \Longrightarrow x_1 = 1,$$
$$y_2 > 0 \Longrightarrow x_2 + x_3 + x_4 = 1,$$
$$y_3 > 0 \Longrightarrow x_2 = 1,$$
$$y_4 > 0 \Longrightarrow x_1 + x_3 + x_5 = 1,$$
$$y_5 > 0 \Longrightarrow x_4 + x_5 + x_6 = 1,$$
$$y_6 > 0 \Longrightarrow x_6 = 1.$$

Az előkészületek után oldjuk meg Edmonds és Johnson eljárásával a 3.3.2. példában megadott feladatot. Az $\mathbf{x}, \mathbf{y}, \mathbf{z}$ vektorváltozóknak az eljárás során kapott értékeit a 3.3.1. táblázatban adjuk meg.

Eljárás.

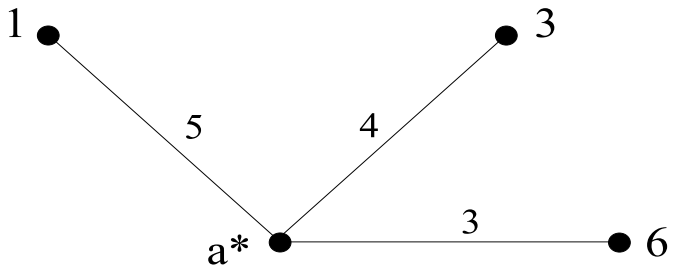
Előkészítő rész. $\mathbf{x} = \mathbf{0}$, $M_0 = \emptyset$, $y_i = 3$, $(i = 1, \ldots, 6)$, $\mathbf{z} = \mathbf{0}$, $\mathcal{G}_0 = \mathcal{G}$, $r = 0$.

Iterációs rész. (0-adik iteráció.)

1. lépés. A 4 csúcsot választjuk. Az alternáló fa egyetlen csúcsból, a 4 csúcsból áll, amely *outer* címkét kapott.
2. lépés. $E^* = \{(2, 4)\}$.
3. lépés. A 4 gyökérből indítva a faépítő eljárást, az a $(2, 4)$ javító utat eredményezi.
4. lépés. (Javító út.) $M_1 = \{(2, 4)\}$, $x_3 = 1$, $\mathcal{G}_1 = \mathcal{G}_0$, $r = r + 1 = 1$, és a következő iterációs lépés következik.

Iterációs rész. (1. iteráció.)

1. lépés. Az 5 csúcsot választjuk. Az alternáló fa ebből az egyetlen csúcsból áll, amelynek *outer* címkéje van.
2. lépés. $E^* = \{(2, 4)\}$.
3. lépés. Az 5 gyökérből indítva a faépítő eljárást, az magyar fát szolgáltat.
5. lépés. (Magyar fa.) $I_1 = \{(2, 5), (4, 5), (5, 6)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{5\}$, $\delta_4 = 3$, $\delta = \min\{\delta_1, \delta_2, \delta_3, \delta_4\} = 1$. Most az (a) szabály szerint kell megváltoztatni $\mathbf{y}$-t, konkrétan y_5 értékét kell csökkenteni 1-gyel. (A változók új értékeit a 3.3.1. táblázat harmadik sorában adtuk meg.)
2. lépés. $E^* = \{(2, 4), (2, 5), (4, 5)\}$.
3. lépés. Az induló fa az 5 csúcsból áll, amelynek *outer* címkéje van. A faépítés során E^*-ból a $(2, 5)$ élet választva, a 2 csúcs *inner* címkét és a 4 csúcs *outer* címkét kap, és a $(2, 5), (2, 4)$ élek *igen* címkét kapnak E^*-ban. Ezek után a $(4, 5)$ él köt össze két *outer* címkéjű csúcsot, így a faépítés páratlan élszámú kört eredményez.
6. lépés. (Páratlan élszámú kör.) Legyen $r = r + 1 (= 2)$, $\mathcal{C}_2 = \{(2, 4), (2, 5), (4, 5)\}$. Zsugorítsuk $\mathcal{G}_1$-et a $\mathcal{C}_2$ körrel, legyen a $\mathcal{C}_2$ körnek megfelelő mesterséges csúcs a^*, és $\mathcal{G}_1$ zsugorítása $\mathcal{G}_2$. Továbbá legyen az M_1 által indukált párosítás $\mathcal{G}_2$-ben M_2, most $M_2 = \emptyset$. Az aktuális alternáló fa gyökere legyen a^* *outer* címkével. (A $\mathcal{G}_2$ zsugorítást mutatja a 3.3.3. ábra.)



3.3.3. ábra. A 3.3.2. példa hálózatának a $\mathcal{C}_2$ körrel való zsugorítása.

2. lépés. $E^* = \{(2,4),(2,5),(4,5)\}$.
3. lépés. Az a^* gyökérből indítva az alternáló fa eljárást, az magyar fát eredményez.
5. lépés. (Magyar fa.) A fa egyetlen *outer* címkéjű csúcsból áll, mégpedig a^*-ból. A változók értékének megváltoztatásakor a mesterséges csúcsba zsugorított csúcsokhoz ugyanazt a címkét rendeljük, mint a mesterséges csúcshoz. Így esetünkben a $2,4,5$ csúcsokhoz *outer* címkét rendelünk. Most $I_1 = \{(1,4),(2,3),(5,6)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{2,4,5\}$, $\delta_4 = 2$, és $\delta = 1$. Az (a) szabálynak megfelelően csökkentsük az y_2, y_4, y_5 változók értékét 1-gyel, majd a (c) szabály szerint növeljük z_{14} értékét 2δ-val, azaz 2-vel. A változók új értékei a 3.3.1. táblázat ötödik sorában találhatók.
2. lépés. $E^* = \{(1,4),(2,4),(2,5),(4,5)\}$.
3. lépés. A fa gyökere az a^* csúcs *outer* címkével. Véve az $(1,a^*)_{14}$ élet, javító utat kapunk.
4. lépés. (Javító út.) Legyen $M_3 = \{(1,a^*)_{14}\}$. $\mathcal{G}_3 = \mathcal{G}_2$, $r = r+1$.

Iterációs rész. (3. iteráció.)

1. lépés. A 3 csúcsot választjuk. Az alternáló fa a 3 csúcspontból áll, amelynek címkéje *outer*.
2. lépés. $E^* = \{(1,4),(2,4),(2,5),(4,5)\}$.
3. lépés. Az alternáló fa eljárás magyar fát eredményez.
5. lépés. (Magyar fa.) A fa csak a 3 csúcsot tartalmazza *outer* címkével. Így az a^* mesterséges csúcsnak nincs címkéje, ami azt eredményezi, hogy a $2,4,5$ csúcsoknak nincs címkéje. Ekkor $I_1 = \{(2,3)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{3\}$, $\delta_4 = 3$, $\delta = \min\{\delta_1,\delta_2,\delta_3,\delta_4\} = 1$. Most csak az (a) szabályt kell alkal-

maznunk, azaz az y_3 változó értékét kell csökkenteni 1-gyel. A változók új értékei a 3.3.1. táblázat hetedik sorában találhatók.

2. lépés. $E^* = \{(1,4),(2,3),(2,4),(2,5),(4,5)\}$.

3. lépés. A fa gyökere a 3 csúcs. A $(3,a^*)_{23}$ élet választva, a^* *inner* címkét kap, és az 1 csúcs *outer* címkét kap, továbbá a $(2,3)$ és $(1,4)$ élek címkét kapnak E^*-ban. Ezzel a faépítés befejeződik, magyar fát eredményez.

5. lépés. (Magyar fa.) Mivel most a^*-nak *inner* címkéje van, ezért a $2,4,5$ csúcsok mindegyikének szintén *inner* címkéje van. Ekkor $I_1 = \emptyset$ $\delta_1 = \infty$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \{14\}$, $\delta_3 = z_{14}/2 = 1$, $I_4 = \{1,3\}$, $\delta_4 = 2$. Ekkor $\delta = 1$. Az (a) szabály szerint csökkentjük az y_1 és y_3 változók értékeit 1-gyel; (b) alapján növeljük y_2, y_4, y_5 értékét 1-gyel, majd (d)-nek megfelelően csökkentjük z_{14} értékét 2-vel. (A változók aktuális értékeit a 3.3.1. táblázat nyolcadik sora tartalmazza.) Mivel a $\delta = \delta_3$ esetet kaptuk, ezért vegyük $\mathcal{G}_3$ a^*-ra vonatkozó rekonstrukcióját, amit jelöljön $\mathcal{G}_4$. ($\mathcal{G}_4$ megegyezik a 3.3.2. ábrán megadott eredeti hálózattal.) Továbbá képezni kell az $M_3 = \{(1,a^*)_{14}\}$ által generált $M_4 = \{(1,4),(2,5)\}$ párosítást $\mathcal{G}_4$-ben. Ezt követően növeljük r értékét 1-gyel és folytassuk az eljárást a kövtkező iterációs lépéssel. (A változók aktuális értékeit a 3.3.1. táblázat kilencedik sora tartalmazza.)

Iterációs rész. (4. iteráció.)

1. lépés. A 6 csúcsot választjuk.

2. lépés. $E^* = \{(1,4),(2,3),(2,4),(2,5),(4,5)\}$.

3. lépés. A 6 gyökérből csak az egyelemű fát tudjuk felépíteni, azaz magyar fát kapunk.

5. lépés. (Magyar fa.) $I_1 = \{(5,6)\}$, $\delta_1 = 2$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{6\}$, $\delta_4 = 3$. Akkor $\delta = \delta_2 = 2$, és (a) szerint csökkentjük y_6 értékét 2-vel. (Az új értékek a 3.3.1. táblázat tizedik sorában találhatók.)

2. lépés. $E^* = \{(1,4),(2,3),(2,4),(2,5),(4,5),(5,6)\}$.

3. lépés. Folytatjuk az alternáló fa építését a 6 gyökérből. Az E^*-ból az $(5,6)$ és $(2,5)$ éleket választjuk, az 5 csúcs *inner*, a 2 csúcs *outer* címkét kap, a tekintett két él pedig *igen* címkével lesz ellátva E^*-ban. Majd a $(2,3)$ élet választva, a $(2,3),(2,5),(5,6)$ élekből álló javító utat kapjuk.

4. lépés. (Javító út.) Legyen $M_5 = \{(1,4),(2,3),(5,6)\}$, változtassuk x_1, x_2, x_6 értékét 1-re és x_4 értékét 0-ra. Legyen $\mathcal{G}_5 = \mathcal{G}_4$.

Növeljük r értékét 1-gyel és térjünk rá a következő iterációra.

Iterációs rész. (5. iteráció.)

1. lépés. Nincs M_5-re nézve szabad csúcs $\mathcal{G}_5$-ben.

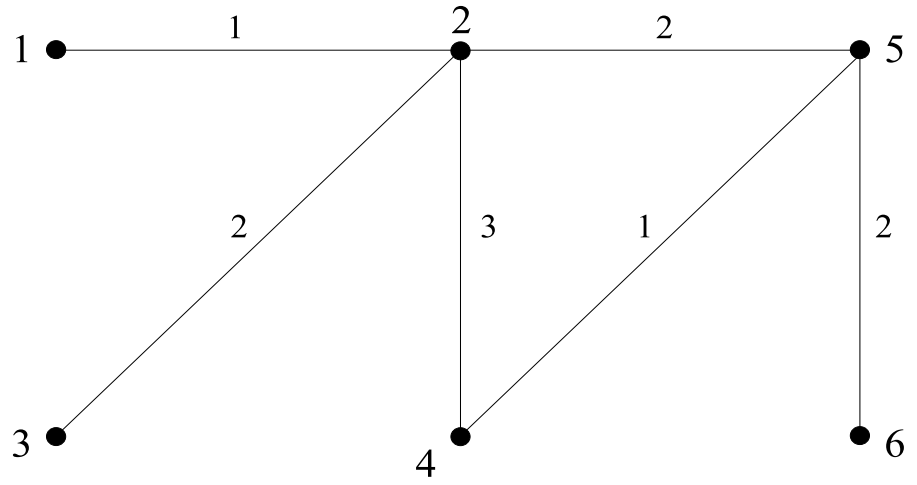
7. lépés. Nincs mesterséges csúcs $\mathcal{G}_5$-ben, ezért vége az eljárásnak, M_5 egy maximális súlyú párosítása $\mathcal{G}$-nek.

	r	x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	y_3	y_4	y_5	y_6	z_{14}
inicial.	0	0	0	0	0	0	0	3	3	3	3	3	3	0
jav. út	1	0	0	1	0	0	0	3	3	3	3	3	3	0
magyar fa	1	0	0	1	0	0	0	3	3	3	3	2	3	0
zsugorítás	2	0	0	1	0	0	0	3	3	3	3	2	3	0
magyar fa	2	0	0	1	0	0	0	3	2	3	2	1	3	2
jav. út	3	0	0	1	0	0	0	3	2	3	2	1	3	2
magyar fa	3	0	0	1	0	0	0	3	2	2	2	1	3	2
magyar fa	3	0	0	1	0	0	0	2	3	1	3	2	3	0
rekonstr.	4	1	0	0	1	0	0	2	3	1	3	2	3	0
magyar fa	4	1	0	0	1	0	0	2	3	1	3	2	1	0
jav. út	5	1	1	0	0	0	1	2	3	1	3	2	1	0

3.3.1. táblázat. Az eljárás alatt a változók értékei.

A fejezet befejezéseként megoldunk két olyan párosítási feladatot, amelyek az alfejezet címében szerepelnek, azaz minimális súlyú teljes párosítást keresünk tetszőleges gráfban. A két demonstrációs példa közül az elsőről kiderül majd, hogy nincs lehetséges megoldása, a második példára pedig meghatározzuk annak egy minimális súlyú teljes párosítását. A továbbiakban csak olyan feladatokat vizsgálunk, amelyekre $|V| = 2n$.

3.3.3. példa Legyen $\mathcal{G} = (\{1, \ldots, 6\}, E)$, ahol $E = \{(1,2), (2,3), (2,4), (2,5), (4,5), (5,6)\}$. Sorszámozzuk az éleket felsorolásuk sorrendjében, és legyenek az élek súlyai ebben a sorrendben $1, 2, 3, 2, 1, 2$. Határozzuk meg a hálózat egy minimális súlyú teljes párosítását! (A hálózat grafikus reprezentációját adja meg a 3.3.4. ábra.)



3.3.4. ábra. A 3.3.3. példa hálózatának grafikus reprezentációja.

A 3.3.1. segédtétel előkészítésének megfelelően, legyen W a súlyok összege plusz 1, azaz $W = 12$, és minden $(i, j) \in E$-re legyen $w'_{ij} = W - w_{ij}$. A 3.3.1. segédtétel szerint, ha tekintjük azt az új hálózatot, amelynek gráfja $\mathcal{G}$ és súlyai a w'_{ij} $((i, j) \in E)$ értékek,

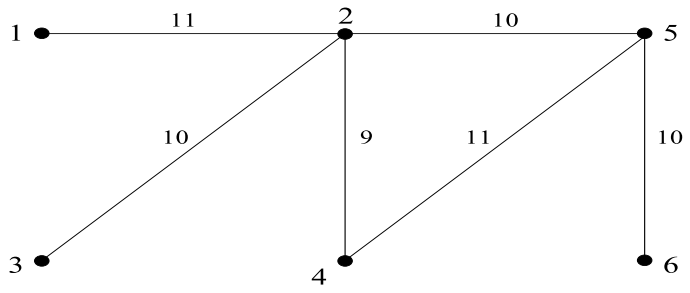
(1) akkor az új hálózatban kell keresni egy maximáli súlyú M_0 párosítást,

(2) ha M_0 súlya nem kisebb, mint $(n-1)W + 1 = 25$, akkor M_0 egy optimális megoldása a 3.3.3. példában megadott feladatnak,

(3) ha M_0 súlya kisebb, mint 25, akkor a 3.3.3. példában megadott feladatnak nincs lehetséges megoldása (nincs $\mathcal{G}$-ben teljes párosítás).

Következésképpen, ha az új hálózatra alkalmazzuk az Edmonds és Johnson féle eljárást, akkor az azzal kapott optimális megoldás segítségével meg tudjuk oldani a 3.3.3. példában kitűzött párosítási feladatot.

Annak érdekében, hogy az eljárás könnyebben követhető legyen, a 3.3.5. ábrán megadjuk az új hálózat grafikus reprezentációját, továbbá az eljárás végrehajtásának részletes leírását. Ezen kívül a 3.3.2. táblázatban megadjuk, hogy miként változnak a változók értékei az eljárás végrehajtása során.



3.3.5. ábra. A 3.3.3. példa hálózatához készített új hálózat grafikus reprezentációja.

Eljárás.

Előkészítő rész. $\mathbf{x} = \mathbf{0}$, $M_0 = \emptyset$, $y_i = 11/2$, $\mathbf{z} = \mathbf{0}$, $\mathcal{G}_0 = \mathcal{G}$, $r = 0$.

Iterációs rész. (0-adik iteráció.)

1. lépés. Az 1 csúcsot választjuk, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,2),(4,5)\}$.
3. lépés. A faépítő eljárás az $(1,2)$ javító utat eredményezi.
4. lépés. (Javító út.) $M_1 = \{(1,2)\}$, $x_1 = 1$, $\mathcal{G}_1 = \mathcal{G}_0$, $r = r+1 = 1$, és a következő iterációs lépés következik.

Iterációs rész. (1. iteráció.)

1. lépés. A 4 csúcsot választjuk, ez a fa gyökere *outer* címkével.
2. lépés. $E^*\{(1,2),(4,5)\}$,
3. lépés. A faépítő eljárás a $(4,5)$ javító utat eredményezi.
4. lépés. (Javító út.) $M_2 = \{(1,2),(4,5)\}$, $x_5 = 1$, $\mathcal{G}_2 = \mathcal{G}_1$, $r = r+1 = 2$, és a következő iterációs lépés következik.

Iterációs rész. (2. iteráció.)

1. lépés. A 3 csúcsot választjuk, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,2),(4,5)\}$,
3. lépés. A faépítő eljárás a $\{3\}$ magyar fát eredményezi.
5. lépés. (Magyar fa.) $I_1 = \{(2,3)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{3\}$, $\delta_4 = 11/2$. Ekkor $\delta = \delta_1 = 1$, y_3 értékét csökkentjük 1-gyel.
2. lépés. $E^* = \{(1,2),(4,5),(2,3)\}$.
3. lépés. A faépítő eljárás a $(2,3),(1,2)$ élekből álló magyar fát adja.
5. lépés. (Magyar fa.) $I_1 = \emptyset$, $\delta_1 = \infty$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{1,3\}$, $\delta_4 = 9/2$. Ekkor $\delta = \delta_4 = 9/2$, csökkentsük y_1 és y_3 értékeit $9/2$-vel, növeljük y_2 értékét $9/2$-vel. Mivel $\delta = \delta_4$, de

az alternáló fa gyökere 3 és y_3 értéke válik 0-vá, ezért $\mathbf{x}$-et nem változtatjuk. $\mathcal{G}_3 = \mathcal{G}_2$, $M_3 = M_2$, $r = r + 1 = 3$, és a következő iterációs lépés következik.

Iterációs rész. (3. iteráció.)

1. lépés. A 6 csúcsot választjuk, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,2),(2,3),(4,5)\}$.
3. lépés. A faépítő eljárás a $\{6\}$ magyar fát eredményezi.
5. lépés. (Magyar fa.) $I_1 = \{(5,6)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{6\}$, $\delta_4 = 11/2$. $\delta = \delta_1 = 1$. Csökkentjük y_6 értékét 1-gyel.
2. lépés. $E^* = \{(1,2),(2,3),(4,5),(5,6)\}$.
3. lépés. A faépítő eljárás az $(5,6),(4,5)$ élekből álló magyar fát adja.
5. lépés. (Magyar fa.) $I_1 = \{(2,4)\}$, $\delta_1 = 13/2$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{4,6\}$, $\delta_4 = 9/2$. $\delta = \delta_4 = 9/2$. Csökkentsük y_4 és y_6 értékeit 9/2-vel, és növeljük y_5 értékét 9/2-vel. Most bár $\delta = \delta_4$, nem változtatjuk $\mathbf{x}$-et. $\mathcal{G}_4 = \mathcal{G}_3$, $M_4 = M_3$, $r = r + 1 = 4$.

Iterációs rész. (4. iteráció.)

1. lépés. Nincs olyan M_4-re nézve szabad i csúcs, amelyre $y_i > 0$ teljesül. Tehát vége az eljárásnak az $M_4 = \{(1,2),(4,5)\}$ párosítás egy maximális súlyú párosítása a tekintett párosítási feladatnak.

	r	x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	y_3	y_4	y_5	y_6	$\mathbf{z}$
inicial.	0	0	0	0	0	0	0	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	**0**
jav. út	1	1	0	0	0	0	0	$\frac{11}{2}$	$\frac{1}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	**0**
jav. út	2	1	0	0	0	1	0	$\frac{11}{2}$	$\frac{1}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	**0**
magyar fa	2	1	0	0	0	1	0	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{9}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	**0**
magyar fa	3	1	0	0	0	1	0	1	10	0	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{11}{2}$	**0**
magyar fa	3	1	0	0	0	1	0	1	10	0	$\frac{11}{2}$	$\frac{11}{2}$	$\frac{9}{2}$	**0**
magyar fa	4	1	0	0	0	1	0	1	10	0	1	10	0	**0**

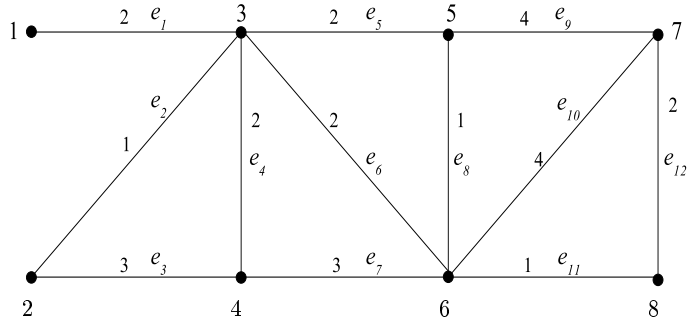
3.3.2. táblázat. A változók értékei az eljárás során.

A fenti táblázat utolsó sorában megadott vektorokról egyszerűen ellenőrizhető, hogy olyan primál-duál megoldáspár, amelyik kielégíti a komplementaritási feltételeket.

Most visszatérve a 3.3.3. feladat megoldására, a feladathoz konstruált, maximális súlyú párosítást kereső párosítási feladat optimuma 22, ami kisebb, mint 25, így a 3.3.1. segédtétel szerint a 3.3.3. feladatnak nincsen lehetséges megoldása (teljes párosítása).

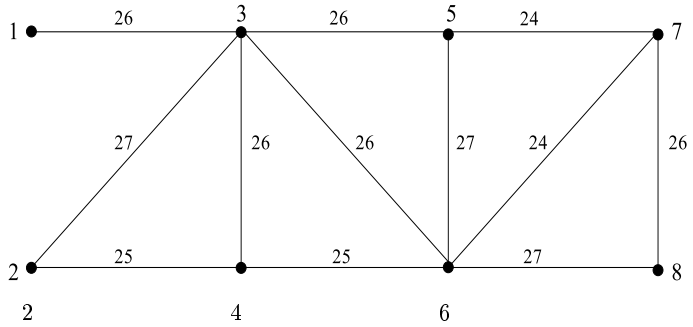
Utolsó példánkban minimális súlyú teljes párosítást keresünk, és meg is határozunk egy ilyen párosítást.

3.3.4. példa. Legyen $\mathcal{G} = (\{1, \ldots, 8\}, E)$, ahol $E = \{(1,3), (2,3), (2,4), (3,4), (3,5), (3,6), (4,6), (5,6), (5,7), (6,7), (6,8), (7,8)\}$. $\mathcal{G}$ élei az adott felsorolás szerint vannak sorszámozva, és a sorszámozás szerint a súlyaik rendre: 2, 1, 3, 2, 2, 2, 3, 1, 4, 4, 1, 2. Határozzuk meg $\mathcal{G}$ egy minimális súlyú teljes párosítását! (A példa hálózatának grafikus reprezentációját a 3.3.6. ábrán adtuk meg.)



3.3.6. ábra. A 3.3.4. példa hálózatának grafikus reprezentációja.

Az előzőekben alkalmazott eljárást követve, $W = 28$, és az élek súlyai az új hálózatban rendre: 26, 27, 25, 26, 26, 26, 25, 27, 24, 24, 27, 26. (A 3.3.7. ábra adja meg az új hálózat grafikus reprezentációját.)



3.3.7. ábra. A 3.3.4. példa hálózatához készített új hálózat grafikus reprezentációja.

Az új hálózatban maximális súlyú párosítást kell keresnünk az Edmonds és Johnson féle eljárással.

Eljárás.

Előkészítő rész. $\mathbf{x} = \mathbf{0}$, $M_0 = \emptyset$, $y_i = 27/2$, $\mathbf{z} = \mathbf{0}$, $\mathcal{G}_0 = \mathcal{G}$, $r = 0$.

Iterációs rész. (0-adik iteráció.)

1. lépés. Az 1 csúcsot választjuk, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(2,3),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás az $\{1\}$ magyar fát eredményezi.
5. lépés. (Magyar fa.) $I_1 = \{(1,3)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{1\}$, $\delta_4 = 27/2$. Így $\delta = \delta_1 = 1$, és y_1 értékét csökkentjük 1-gyel.
2. lépés. $E^* = \{(1,3),(2,3),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás az $(1,3)$ javító utat adja.
4. lépés. (Javító út.) $M_1 = \{(1,3)\}$, $x_1 = 1$, $\mathcal{G}_1 = \mathcal{G}_0$, $r = r+1 = 1$, és a következő iterációs lépés következik.

Iterációs rész. (1. iteráció.)

1. lépés. Válasszuk a 2 csúcsot, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,3),(2,3),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás a $(2,3),(1,3)$ élekből álló magyar fát eredményezi.
5. lépés. (Magyar fa.) $I_1 = \{(2,4)\}$, $\delta_1 = 2$, $I_2 = \emptyset$, $\delta_2 = \infty$, $I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{1,2\}$, $\delta_4 = 27/2$. Így $\delta = \delta_1 = 2$, y_1 és y_2 értékeit csökkentjük 2-vel, és y_3 értékét növeljük 2-vel.
2. lépés. $E^* = \{(1,3),(2,3),(2,4),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás a $(2,4)$ javító utat adja.
4. lépés. (Javító út.) $M_2 = \{(1,3),(2,4)\}$, $x_3 = 1$, $\mathcal{G}_2 = \mathcal{G}_0$, $r = r+1 = 2$, és a következő iterációs lépés következik.

Iterációs rész. (2. iteráció.)

1. lépés. Válasszuk az 5 csúcsot, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,3),(2,3),(2,4),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás az $(5,6)$ javító utat adja.
4. lépés. (Javító út.) $M_3 = \{(1,3),(2,4),(5,6)\}$, $x_8 = 1$, $\mathcal{G}_3 = \mathcal{G}_0$, $r = r+1 = 3$, és a következő iterációs lépés következik.

Iterációs rész. (3. iteráció.)

1. lépés. Válasszuk a 7 csúcsot, ez a fa gyökere *outer* címkével.
2. lépés. $E^* = \{(1,3),(2,3),(2,4),(5,6),(6,8)\}$.
3. lépés. A faépítő eljárás a $\{7\}$ magyar fát adja.
5. lépés. (Magyar fa.) $I_1 = \{((5,7),(6,7),(7,8)\}$, $\delta_1 = 1$, $I_2 = \emptyset$, $\delta_2 = \infty$,

$I_3 = \emptyset$, $\delta_3 = \infty$, $I_4 = \{7\}$, $\delta_4 = 27/2$. Így $\delta = \delta_1 = 1$, y_7 értékét csökkentjük 1-gyel.

2. lépés. $E^* = \{(1,3),(2,3),(2,4),(5,6),(6,8),(7,8)\}$.

3. lépés. A faépítő eljárás a $(7,8)$ javító utat adja.

4. lépés. (Javító út.) $M_4 = \{(1,3),(2,4),(5,6),(7,8)\}$, $x_{12} = 1$, $\mathcal{G}_4 = \mathcal{G}_0$, $r = r + 1 = 4$, és a következő iterációs lépés következik.

Iterációs rész. (4. iteráció.)

1. lépés. Nincs M_4-re nézve szabad csúcs $\mathcal{G}_4$-ben, ezért vége az eljárásnak, az M_4 párosítás maximális súlyú párosítása a tekintett feladatak.

A változók aktuális értékei: $x_1 = x_3 = x_8 = x_{12} = 1$, $x_t = 0$ a többi indexre, $\mathbf{z} = \mathbf{0}$, $y_1 = 21/2$, $y_2 = 23/2$, $y_3 = 31/2$, $y_7 = 25/2$ és $y_t = 27/2$ a kimaradó indexekre.

Most a kritikus komplementaritási feltételek ((ii) feltételcsoport) a következők:

$$
\begin{aligned}
&y_1 > 0 \Longrightarrow (x_1 = 1)\\
&y_2 > 0 \Longrightarrow (x_2 + x_3 = 1)\\
&y_3 > 0 \Longrightarrow (x_1 + x_2 + x_4 + x_5 + x_6 = 1)\\
&y_4 > 0 \Longrightarrow (x_3 + x_4 + x_7 = 1)\\
&y_5 > 0 \Longrightarrow (x_5 + x_8 + x_9 = 1)\\
&y_6 > 0 \Longrightarrow (x_6 + x_7 + x_8 + x_{10} + x_{11} = 1)\\
&y_7 > 0 \Longrightarrow (x_9 + x_{10} + x_{12} = 1)\\
&y_8 > 0 \Longrightarrow (x_{11} + x_{12} = 1)
\end{aligned}
$$

Egyszerűen ellenőrizhető, hogy az adott $\mathbf{x}$-re a jobboldali egyenletek mindegyike teljesül.

Most visszatérve a kiindulási 3.3.4. példára, az eljárással meghatározott M_4 párosítás súlya 104. Másrészt $W = 28$, és így $(n-1)W + 1 = 85$ kisebb, mint a kapott 104 optimumérték. Ekkor a 3.3.1. segédtétel alapján az M_4 párosítás egy minimális súlyú teljes párosítása a 3.3.4. példa hálózatának. M_4 súlya a kiindulási példában: 8.

A fejezet befejezéseként megemlítjük, hogy a kutatók a megismert két algoritmus különböző változtatásaival, módosításaival, fejlesztéseivel, olyan új algoritmusokat dolgoztak ki, amelyek hatékonyabbak, mint az általunk bemutatott eljárások. A teljesség igénye nélkül ilyen jellegű eredményeket tartalmaznak a következő munkák: [64], [104], [122], [141], [179].

4. Legrövidebb utak hálózatokban

A jelen fejezetben feltételezzük, hogy a hálózattal egy úthálózatot adunk meg, és az élekhez rendelt súlyok rendre az illető útszakasz hosszát adják meg. Ekkor gyakorlati szempontból is igen fontos probléma a hálózat egy előírt kiindulási pontjából egy előírt végpontba vezető, legrövidebb útjának a meghatározása. A fenti problémát szokásos a *legrövidebb út problémájának* nevezni.

Megjegyezzük, hogy az úthálózat nem okvetlen alkot irányítatlan gráfot, például gondoljunk arra, hogy valamely útszakaszon félútlezárás miatt csak az egyik irányban lehet közlekedni, vagy elterelés miatt i-ből j-be hosszabb az út, mint j-ből i-be. A felsoroltak miatt a jelen fejezetben gráfon mindig irányított gráfot értünk.

A továbbiakban az egyszerűbb tárgyalás érdekében rendre feltételezzük, hogy a tekintett hálózat csúcspontjai az $1, \ldots, n$ természetes számok, és a kiindulási csúcs 1, továbbá a végpont n. A hálózat mátrix-reprezentációját úgy definiáljuk, hogy $W = +\infty$, azaz a nem létező élekhez a $+\infty$ mennyiséget rendeljük. Ez gyakorlatilag egy nagy gépi számnak tekinthető, melyről feltételezzük, hogy valamilyen konstanst hozzáadva vagy kivonva belőle ez a mennyiség nem változik.

A probléma megoldására különböző hatékonyságú eljárások kerültek kidolgozásra attól függően, hogy a hálózat milyen tulajdonságokkal rendelkezik. A továbbiakban az alábbi tulajdonságok mellett fogunk megadni eljárásokat.

(1) A hálózat nem tartalmaz negatív hosszúságú kört.

(2) A hálózatban minden élhossz pozitív.

(3) A hálózat nem tartalmaz kört.

Fontosnak tartjuk megjegyezni, hogy az általunk megadásra kerülő eljárásokon kívül még számos további eljárás is van, ráadásul más jellegű, a legrövidebb utakhoz kapcsolódó problémák is a vizsgálatok tárgyát képezik. A téma iránt érdeklődőknek ajánljuk a [42], [54], [66], [122] munkákat.

Az (1) feltétel mellett egymástól függetlenül R. E. Bellman [18] és L. R. Ford [57] dolgozta ki az alábbi iterációs eljárást, amely *fokozatos közelítés módszere* néven ismeretes.

Fokozatos közelítés módszere ([18], [57])

Előkészítő rész

Legyen $u_1^{(1)} = 0$, $u_j^{(1)} = d_{1j}$, $j = 2, \ldots, n$, továbbá adjuk r-nek az 1 értéket.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $r = n - 1$, akkor vége az eljárásnak, $u_j^{(n-1)}$ az 1 csúcsból a j csúcsba vezető legrövidebb út hossza. Ha $r < n - 1$, akkor térjünk rá a 2. lépésre.

- *2. lépés.* Rendre számítsuk ki az $u_j^{(r+1)}$, $j = 1, \ldots, n$ értékeket a következők szerint:

$$u_j^{(r+1)} = \min\{u_j^{(r)}, \min_{k \neq j}\{u_k^{(r)} + d_{kj}\}\}.$$

  Növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

Az eljárás helyességének igazolásához m szerinti teljes indukcióval bizonyítjuk, hogy $u_j^{(m)}$ az 1 csúcsból a j csúcsba vezető, legfeljebb m élet tartalmazó utak közül a legrövidebb út hossza.

Ha $m = 1$, akkor az állítás nyilvánvalóan igaz az $u_j^{(1)}$, $j = 1, \ldots, n$ értékek definíciójából. Ezek után legyen $1 \leq m < n - 1$ és tegyük fel, hogy az állítás teljesül m-re. Legyen $j \in \{1, \ldots, n\}$ tetszőleges.

Tekintsük az 1 csúcsból a j csúcsba vezető, legfeljebb $m+1$ élet tartalmazó utakat, ahol figyelembe vesszük az olyan utakat is, amelyek nemlétező éleket tartalmaznak, az ilyen utak hossza $+\infty$. Mivel véges sok ilyen út létezik, ezért ezek között vannak legkisebb hosszúságúak. Legyen egy legkisebb hosszúságú, legfeljebb $m + 1$ élet tartalmazó útban a j csúcs őse l. Akkor az indukciós feltevés alapján ezen út hossza $u_l^{(m)} + d_{lj}$. Másrészt

$$u_j^{(m+1)} = \min\{u_j^{(m)}, \min_{k \neq j}\{u_k^{(m)} + d_{kj}\}\} \leq u_l^{(m)} + d_{lj}.$$

Most vegyük észre, hogy a fenti egyenlőtlenségben az egyenlőségnek kell teljesülnie, mivel ellenkező esetben az indukciós feltevés alapján azt kapnánk, hogy létezik az $u_l^{(m)} + d_{lj}$ hossznál rövidebb, legfeljebb $m+1$ élet tartalmazó 1-ből j-be vezető út, ami ellentmond annak, hogy a tekintett út egy legrövidebb út. Tehát, az 1-ből j-be vezető, legfeljebb $m+1$ élet tartalmazó utak közül a legrövidebbek hossza $u_j^{(m+1)}$.

Mivel bármely 1-ből j-be vezető út legfeljebb $n-1$ élet tartalmazhat, ugyanis az út definíciójában kikötöttük, hogy az általa meghatározott csúcssorozat páronként különböző csúcsokat tartalmaz, ezért a bizonyított állítás alapján $u_j^{(n-1)}$ az ilyen utak közül a minimális hosszúságúnak a hosszát adja meg. (Speciálisan, ha $u_j^{(n-1)} = +\infty$, akkor ez azt jelenti, hogy a gráfban nem létezik az 1-ből j-be vezető út, vagy másképp fogalmazva minden ilyen út tartalmaz legalább egy nemlétező élet, amihez a $+\infty$-t rendeltük.) Ezzel az eljárás helyességét igazoltuk.

A legrövidebb út hosszának ismeretében könnyen meghatározható a hálózatban egy ilyen hosszúságú út a tekintett csúcsból visszafelé haladva, és minden lépésben olyan élet választva, hogy az aktuális élre $u_k^{(m-1)} + d_{kj} = u_j^{(m)}$ teljesüljön.

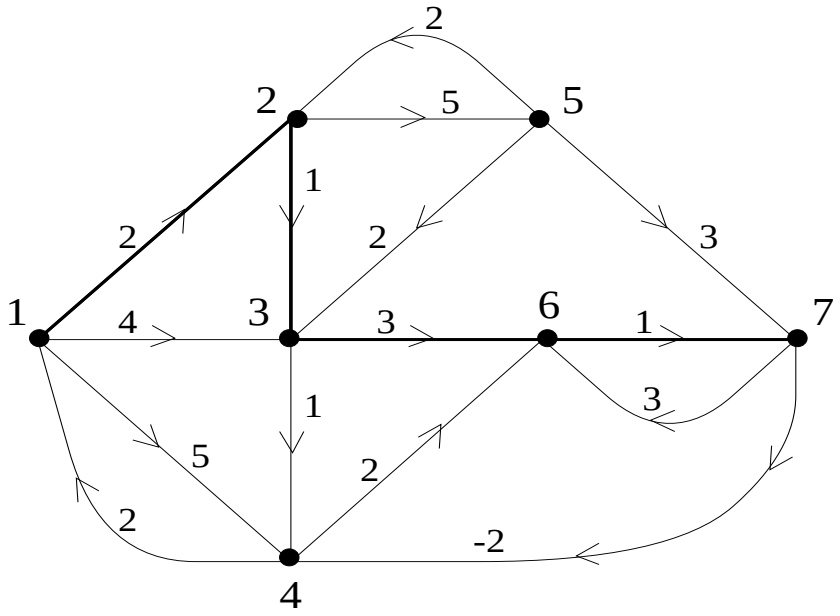
Az eljárással kapcsolatban érdemes megemlíteni, hogy a kezdőpontból a végpontba vezető utak közül a legrövidebb hosszúságú utak hosszának meghatározásához szükséges a kezdőpontból a hálózat minden egyes pontjába vezető utakra a legrövidebb utak hosszát is meghatározni. Ez a jelenség általános, a további eljárásokban is ezt fogjuk majd tapasztalni.

Az $u_j^{(m)}$ értékek definíciójából nyilvánvaló, hogy amennyiben az eljárás során $u_j^{(m-1)} = u_j^{(m)}$, $j = 1, \ldots, n$ teljesül, akkor ezen értékek a további iterációk során nem változnak, azaz megkapjuk a legrövidebb utak hosszát.

Végül megemlítjük, hogy egyszerűen belátható, miszerint a fokozatos közelítés módszerének műveletigénye $O(n^3)$.

Az eljárást a következő példán illusztráljuk.

4.1. példa. Tekintsük azt a hálózatot, amelynek csúcspontjai: $V = \{1, 2, \ldots, 7\}$, továbbá élei és az élek hosszai a következő mátrix-reprezentációval vannak megadva, ahol a $+\infty$-t egyszerűen ∞-nel jelöltük. A hálózat grafikus reprezentációját a 4.1. ábra tartalmazza.



4.1. ábra. A 4.1. példa hálózatának grafikus reprezentációja.

∞	2	4	5	∞	∞	∞
∞	∞	1	∞	5	∞	∞
∞	∞	∞	1	∞	3	∞
2	∞	∞	∞	∞	2	∞
∞	2	2	∞	∞	∞	3
∞	∞	∞	∞	∞	∞	1
∞	∞	∞	-2	∞	3	∞

$u_j^{(1)}$	0	2	4	5	∞	∞	∞
$u_j^{(2)}$	0	2	3	5	7	7	∞
$u_j^{(3)}$	0	2	3	4	7	6	8
$u_j^{(4)}$	0	2	3	4	7	6	7
$u_j^{(5)}$	0	2	3	4	7	6	7

Az eljárás manuális végrehajtásához praktikus a (d_{ij}) mátrixot és az $u_j^{(m)}$ értékekből álló mátrixot egymás mellett szerepeltetni, mivel $u_j^{(m+1)}$ kiszámításához az $u_k^{(m)}$ $k \neq j$, értékekre van szükség, amelyek a megelőző sorban helyezkednek el, valamint a d_{kj}, $k \in \{1, \ldots, n\} \setminus \{j\}$ értékekre, amelyek a (d_{ij}) mátrix j-edik oszlopában találhatók.

A fenti példában az 1 csúcsból a 7 csúcsba vezető, legrövidebb út hossza 7. Ennek alapján egy legrövidebb utat a következőképpen határozhatunk meg. Tekintsük a 7 csúcs őseit, ezek a jelen példában az 5 és a 6 csúcsok, és az ősök közül kiválasztjuk azt, amelyre az 1 csúcsból az illető ősbe vezető

legrövidebb út hosszának és az ősből a 7 csúcsba vezető él hosszának összege pontosan az 1 csúcsból a 7 csúcsba vezető legrövidebb út hosszát adja. Formálisan, a végpont azon k ősét keressük, amelyre $u_k^{(n-1)} + d_{kn} = u_n^{(n-1)}$ teljesül. A legrövidebb út létezéséből nyilvánvalóan következik, hogy legalább egy ilyen ős létezik. Esetünkben ez a 6 csúcspont. Most a 6 csúcshoz keresve alkalmas őst, két olyan ős is lesz, amelyik kielégíti a feltételt, nevezetesen a 3 és 4 csúcsok. Közülük a 3 csúcsot választva, a 3 alkalmas őseként a 2 csúcsot kapjuk, majd 2 alkalmas őseként adódik az 1 kiindulási csúcs. Tehát egy legrövidebb út az $(1,2),(2,3),(3,6),(6,7)$ élek által meghatározott út, amit vastag vonallal emeltünk ki a hálózat 4.1. ábráján.

A tekintett példa azt is mutatja, hogy a legrövidebb út nem szükségképp egyértelműen meghatározott, a 6 csúcsnak a másik alkalmas ősét választva, az $(1,2),(2,3),(3,4),(4,6),(6,7)$ élekből álló legrövidebb utat kapjuk.

A fokozatos közelítés módszerével kapcsolatban fontos még megjegyezni a következőt. Tetszőleges hálózatra alkalmazva a módszert belátható, hogy a hálózat akkor és csak akkor tartalmaz negatív hosszúságú kört, ha nem teljesülnek az $u_j^{(n-1)} = u_j^{(n)}$, $j = 1, \ldots, n$ egyenlőségek. Ezen észrevétel alapján az eljárás használható annak eldöntésére, hogy a tekintett hálózat tartalmaz-e negatív hosszúságú kört.

Következőként azt az esetet vizsgáljuk, amikor a hálózat kielégíti a (2) feltételt, azaz a hálózatban minden élhossz pozitív.

A tekintett esetre egy E. W. Dijkstra-tól [44] származó, $O(n^2)$ műveletigényű iterációs eljárással fogunk megismerkedni. Az eljárás azon alapul, hogy a csúcspontok halmazát felírjuk két diszjunkt részhalmaz egyesítéseként, ezek az r-edik iterációs lépésben I_r és J_r. Az I_r halmaz tartalmazza azon csúcsokat, amelyekre már ismerjük az 1 kezdőpontból az illető csúcsokba vezető legrövidebb utak hosszát; J_r pedig azon csúcspontok halmaza, amelyekre a kezdőpontból a hozzájuk vezető, legrövidebb út hossza még nem ismeretes. Az iteráció során J_r-ből törölünk egy pontot, a törlés utáni halmaz lesz J_{r+1}, továbbá a törölt ponttal bővítjük az I_r halmazt, a bővített halmaz lesz I_{r+1}. Nyilvánvalóan, amennyiben minden iterációs lépésben tudunk úgy választani pontot J_r-ből, hogy az új pontra meghatározható legyen a kezdőpontból a hozzá vezető, legrövidebb út hossza, akkor n iterációs lépésben már minden pontra ismert lesz a kezdőpontból az illető pontba vezető legrövidebb út hossza. A vázolt technikát a következő eljárás valósítja meg.

Dijkstra módszere ([44])

Előkészítő rész

Legyen $u_1 = 0$, $u_j = d_{1j}$, $j = 2, \ldots, n$, $I_1 = \{1\}$ és $J_1 = \{2, \ldots, n\}$, továbbá adjuk r-nek az 1 értéket.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Keressünk olyan $k \in J_r$ csúcsot, amelyre $u_k = \min_{j \in J_r} u_j$ teljesül. Legyen $I_{r+1} = I_r \cup \{k\}$ és $J_{r+1} = J_r \setminus \{k\}$. Ha $J_{r+1} = \emptyset$, akkor vége az eljárásnak; különben folytassuk az eljárást a 2. lépéssel.

- *2. lépés.* Minden $j \in J_{r+1}$ egész számra változtassuk meg u_j értékét a következők szerint. Legyen

$$u_j = \min\{u_j, u_k + d_{kj}\}.$$

  Növeljük r értékét 1-gyel és térjünk rá a következő iterációra.

Az eljárás helyességének igazolásához r szerinti teljes indukcióval igazoljuk az I_r és J_r halmazok alábbi tulajdonságait:

(a) minden $i \in I_r$ csúcsra u_i az 1 csúcsból az i csúcsba vezető, legrövidebb út hosszát adja meg, és az ilyen legrövidebb utak között van olyan, amelynek minden csúcsa I_r-beli.

(b) minden $j \in J_r$ csúcsra u_j az 1 pontból a j pontba vezető, és j kivételével csak I_r-beli csúcsokat tartalmazó utak közül egy legrövidebb ilyen út hossza.

Ha $r = 1$, akkor az (a) és (b) állítások mindegyike nyilvánvalóan teljesül. Most tegyük fel, hogy $1 \leq r < n$, és az I_r, J_r halmazokra teljesülnek a fenti állítások. Legyen $u_k = \min_{j \in J_r} u_j$.

Elsőként megmutatjuk, hogy az $I_{r+1} = I_r \cup \{k\}$ halmazra teljesül az (a) állítás. Ehhez legyen $i \in I_{r+1}$ tetszőleges. Ha $i \neq k$, akkor $i \in I_r$ és az indukciós feltevés alapján azt kapjuk, hogy u_i az 1 csúcsból az i csúcsba vezető legrövidebb út hossza, továbbá van olyan az 1 csúcsból az i csúcsba vezető, legrövidebb út, amely csak I_r-beli csúcsokat tartalmaz. Következésképp már csak k-ra kell igazolni az (a) állítást. Az indukciós feltevés alapján a (b) állításból következik, hogy létezik az 1 csúcsból a k csúcsba vezető olyan út, amely k-tól eltekintve csak I_r-beli pontokat tartalmaz és u_k egy legrövidebb

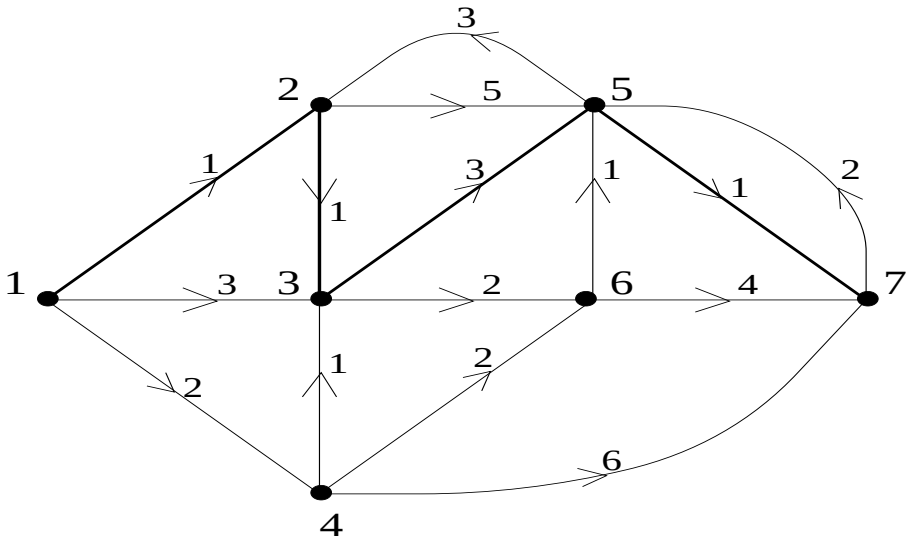
ilyen út hossza. Vegyük észre, hogy az ilyen utak minden csúcsa eleme az I_{r+1} halmaznak. (Megjegyezzük hogy ez az út tartalmazhat nemlétező élet is, ilyenkor az úthossz $+\infty$.) Megmutatjuk, hogy ekkor u_k az 1 csúcsból a k csúcsba vezető legrövidebb út hossza. Ehhez elegendő belátni, hogy tetszőleges, az 1 csúcsból a k csúcsba vezető út hossza nem kisebb, mint u_k. Ha a tekintett út k kivételével csak I_r-beli csúcsokat tartalmaz, akkor az indukciós feltevés alapján (b)-ből következik, hogy a tekintett út hossza nem kisebb, mint u_k. Ha a tekintett út k-n kívül tartalmaz további J_r-beli csúcsokat, akkor mivel a kezdőpont 1 és $1 \in I_r$ az út I_r-beli pontokkal kezdődik és lesz egy első olyan csúcs az úton, ami már nem eleme I_r-nek. Jelölje ezt a pontot j. Akkor nyilvánvalóan $j \in J_r$, továbbá a tekintett út hossza az indukciós feltevés (b) része alapján nem kisebb, mint $u_j + \Delta$, ahol $\Delta > 0$ a j pont után következő szakaszok hosszainak összegét jelöli. Másrészt k definíciójából következik, hogy $u_k \leq u_j$. Ezzel azt kapjuk, hogy $u_j + \Delta$ nem kisebb, mint u_k. Ennél több nem igaz, ugyanis, ha a tekintett utak tartalmaznak nem létező éleket, akkor u_k és u_j a $+\infty$-nel egyenlő. Ezzel igazoltuk, hogy tetszőleges az 1 csúcsból a k csúcsba vezető út hossza nem kisebb, mint u_k, azaz u_k az 1 csúcsból a k csúcsba vezető legrövidebb út hossza. Továbbá az is igazolást nyert, hogy az 1 csúcsból a k csúcsba vezető, u_k hosszúságú utak között van olyan, amely csak I_{r+1}-beli pontokat tartalmaz.

Következő lépésként igazoljuk, hogy a módosított u_j értékek, amelyeket u'_j-vel jelölünk, az 1 pontból a j pontba vezető, és j kivételével csak I_{r+1}-beli csúcsokat tartalmazó utak közül a legrövidebb utak hosszát adják meg minden $j \in J_{r+1}$ egészre. Ehhez legyen $j \in J_{r+1}$ tetszőleges és tekintsünk egy olyan legrövidebb utat, amely j kivételével csak I_{r+1}-beli pontokat tartalmaz. Ha az illető út nem tartalmazza a k csúcsot, akkor az indukciós feltevés (b) állítása alapján hossza pontosan u_j. Ha az illető út tartalmazza a k csúcsot, akkor feltehetjük, hogy j őse k ebben az útban. Valóban, ha j őse valamely $i \neq k$ csúcs lenne, akkor az indukciós feltevés alapján a tekintett út hossza nem lenne kisebb, mint $u_i + d_{ij} = u_j$. Ha j őse k a tekintett útban, akkor ezen út hossza $u_k + d_{kj}$. Következésképp, a tekintett legrövidebb út hossza $\min\{u_j, u_k + d_{kj}\}$, ami definíció szerint pontosan u'_j. Ezzel a (b) állítást is igazoltuk.

Nyilvánvalóan az (a) állításból következik az eljárás helyessége.

Az eljárás illusztrálására megoldjuk a következő feladatot.

4.2. példa. Tekintsük azt a hálózatot, amelynek csúcsai az $1, 2, \ldots, 7$ pozitív egész számok, és amelynek élei valamint az élek hosszai a következő mátrix-reprezentációval adottak, ahol ismét a $+\infty$-t ∞-nel helyettesítettük. A hálózat grafikus reprezentációját mutatja a 4.2 ábra.



4.2. ábra. A 4.2. példa hálózatának grafikus reprezentációja.

							u_1	u_2	u_3	u_4	u_5	u_6	u_7
∞	1	3	2	∞	∞	∞	0	1	3	2	∞	∞	∞
∞	∞	1	∞	5	∞	∞	0	1	2	2	6	∞	∞
∞	∞	∞	∞	3	2	∞	0	1	2	2	5	4	∞
∞	∞	1	∞	∞	2	6	0	1	2	2	5	4	8
∞	3	∞	∞	∞	∞	1	0	1	2	2	5	4	8
∞	∞	∞	∞	1	∞	4	0	1	2	2	5	4	6
∞	∞	∞	∞	2	∞	∞							

A legrövidebb úthosszak ismeretében ismét visszafelé haladva, meghatározhatunk egy legrövidebb utat, amely az 1 kezdőpontból a 7 végpontba vezet. Azt kapjuk, hogy a legrövidebb út nem egyértelmű, nevezetesen az $(1,2),(2,3),(3,5),(5,7)$ és $(1,2),(2,3),(3,6),(6,5),(5,7)$ élsorozatok által meghatározott utak mindegyike 6 hosszúságú legrövidebb út. Közülük az elsőt vastag vonallal jelöltük meg a 4.2. ábrán.

A fejezet folytatásaként most azokat a hálózatokat tekintjük, amelyek kielégítik a (3) feltételt, azaz körmentesek. A legrövidebb út ilyen hálózatokban történő meghatározására szolgál a következő $O(n^2)$ műveletigényű iterációs eljárás, melyről nem lehet tudni, hogy kitől származik. Az eljárás feltételezi, hogy a gráf csúcsai az $1, \ldots, n$ természetes számok, és minden $i \neq j \in \{1, \ldots, n\}$ számpárra, amennyiben (i, j) éle a gráfnak, akkor $i < j$ teljesül. A 2.1. segédtételből tudjuk, hogy körmentes esetben a gráf csúcsai sorszámozhatók úgy, hogy a sorszámokkal helyettesítve a csúcsokat, a fenti tulajdonság érvényes.

Eljárás körmentes hálózatokra

Előkészítő rész

Legyen $u_1 = 0$, továbbá adjuk r-nek az 1 értéket.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $r = n$, akkor vége az eljárásnak, u_n az 1 csúcsból az n csúcsba vezető legrövidebb út hossza. Különben hajtsuk végre a 2. lépést.

- *2. lépés.* Képezzük a

$$\min_{k<r+1} \{u_k + d_{k,r+1}\}$$

mennyiséget, és adjuk u_{r+1}-nek a minimumértéket. Növeljük r értékét 1-gyel és térjünk rá a következő iterációra.

Az eljárás helyességének igazolásához m szerinti teljes indukcióval megmutatjuk, hogy u_m az 1 csúcsból az m csúcsba vezető legrövidebb út hossza.

Ha $m = 1$, akkor az állítás nyilvánvalóan teljesül. Most tegyük fel, hogy $1 \leq m < n$, és az állítás teljesül minden m-nél nem nagyobb pozitív egészre. Tekintsünk egy, az 1 csúcsból az $m + 1$ csúcsba vezető legrövidebb utat. Jelölje $m + 1$ ősét ebben az útban k. Most különböztessünk meg két esetet az u_{m+1} értékétől függően.

1. eset. $u_{m+1} = +\infty$. Megmutatjuk, hogy ebben az esetben minden az 1 csúcsból az $m + 1$ csúcsba vezető út tartalmaz legalább egy nemlétező élet. Ezt indirekt igazoljuk. Ha lenne 1-ből $m + 1$-be vezető véges út, akkor a csúcsok feltételezett tulajdonsága miatt az út $1, i_1, \ldots, i_k, m + 1$ csúcspontjaira az $1 < i_1 < \ldots < i_k < m + 1$ relációknak kellene fennállnia. De akkor szükségképpen kell létezni legalább egy véges útnak 1-ből i_k-ba és

$d_{i_k,m+1} < +\infty$. De akkor $u_{i_k} + d_{i_k,m+1} < +\infty$, amivel azt kapjuk, hogy a fenti minimum véges, ami ellentmondás.

2. eset. $u_{m+1} < +\infty$. Ekkor van olyan k, hogy $k < m+1$, $d_{k,m+1} \neq +\infty$ és $u_{m+1} = u_k + d_{k,m+1}$, azaz létezik egy véges út 1-ből $m+1$-be. Most megmutatjuk, hogy bármely, az 1-ből az $m+1$-be vezető út hossza nem kisebb, mint u_{m+1}. Ehhez tekintsünk egy tetszőleges 1-ből $m+1$-be vezető utat. Ha az út tartalmaz nemlétező élet, akkor a hossza $+\infty$, így elegendő csak olyan utak vizsgálatára szorítkozni, amelyek nem tartalmaznak nemlétező éleket. Legyenek egy ilyen út csúcspontjai $1, j_1, \ldots, j_s, m+1$. Akkor a feltételünk szerint $1 < j_1 < \ldots < j_s < m+1$. Másrészt az indukciós feltevés alapján ezen út hossza nem kisebb, mint $u_{j_s} + d_{j_s,m+1}$. De a j_s index szerepel a fenti minimumképzésben, így $u_k + d_{k,m+1} \leq u_{j_s} + d_{j_s,m+1}$, amivel adódik, hogy a tekintett út hossza nem kisebb, mint $u_k + d_{k,m+1}$, azaz $u_{m+1} = u_k + d_{k,m+1}$ valóban egy, az 1 csúcsból az $m+1$ csúcsba vezető legrövidebb út hossza.

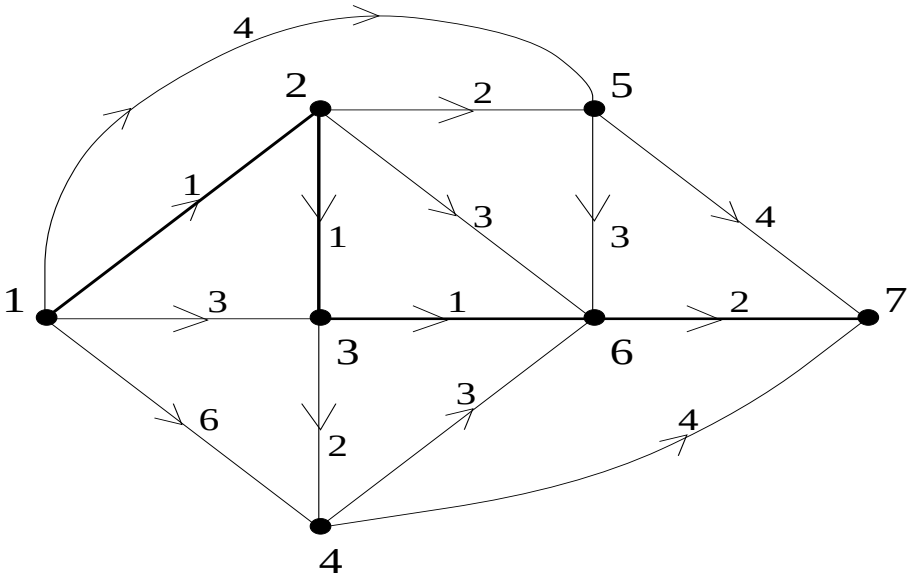
Vegyük észre, hogy a feltételezett tulajdonság mellett a hálózat mátrixreprezentációjában csak a diagonális felett szerepelhetnek véges értékek. Ez az észrevétel hasznos az eljárás végrehajtási technikáját illetően. Nevezetesen praktikus a következő elrendezést használni.

$$\begin{array}{cccccccc}
u_1| & d_{12} & d_{13} & d_{14} & \ldots & d_{1,n-1} & d_{1n} \\
 & u_2| & d_{23} & d_{24} & \ldots & d_{2,n-1} & d_{2n} \\
 & & u_3| & d_{34} & \ldots & d_{3,n-1} & d_{3n} \\
\vdots & & & & \ldots & & \vdots \\
 & & & & \ldots & u_{n-1}| & d_{n-1,n} \\
 & & & & \ldots & & u_n|
\end{array}$$

Ezen elrendezés esetében az u_j érték kiszámításához a tőle balra elhelyezkedő u_k valamint az u_j oszlopában levő d_{kj} értékekre van szükség, ahol $1 \leq k < j$.

Az eljárás illusztrálására tekintsük a következő feladatot.

4.3. példa. Tekintsük azt a hálózatot, amelynek csúcspontjai az $1, \ldots, 7$ pontok, továbbá élei és az élhosszak az alábbi mátrix diagonálisa felett helyezkednek el. A hálózat grafikus reprezentációját a 4.3. ábra adja meg.



4.3. ábra. A 4.3. példa hálózatának grafikus reprezentációja.

Ezen hálózat nyilvánvalóan körmentes. Alkalmazva rá az eljárást, az alábbi mátrixban szereplő u_j értékeket kapjuk.

$$\begin{array}{cccccccc}
0| & 1 & 3 & 6 & 4 & \infty & \infty \\
 & 1| & 1 & \infty & 2 & 3 & \infty \\
 & & 2| & 2 & \infty & 1 & \infty \\
 & & & 4| & \infty & 3 & 4 \\
 & & & & 3| & 3 & 4 \\
 & & & & & 3| & 2 \\
 & & & & & & 5|
\end{array}$$

Visszafelé haladva, ismét meghatározhatunk egy legrövidebb utat, amely a jelen példánál egyértelműen meghatározott. Nevezetesen az $(1,2),(2,3),(3,6),(6,7)$ élekből álló út hossza a legkisebb, 5 egység. Az utat az ábrán vastag vonallal jelöltük meg.

A fejezet befejezéseként egy olyan problémával foglalkozunk, amely szorosan kapcsolódik a legrövidebb utakhoz. A probléma a következő.

k-adik legrövidebb utak problémája

Legyen adott egy hálózat az $1,\ldots,n$ csúcsokkal és egy $1 \leq k \leq n$ egész. Határozzuk meg az 1 csúcsból az n csúcsba vezető utakra az első k legrövidebb úthosszt!

Gyakorlati alkalmazásként például egy légitársaságnak fontos lehet, hogy amennyiben egyik kliense lemaradt vagy lemarad a legrövidebb úton haladó járatról, mi a második legrövidebb úthossz.

Mielőtt továbblépnénk pontosítani kell magát a feladatot. A 2. fejezetben úgy definiáltuk az irányított út fogalmát, hogy a benne szereplő csúcsok páronként különbözők. Ezeket az utakat szokás *egyszerű utaknak* nevezni. Ha nem kötjük ki, hogy az útban levő csúcsok páronként különbözők legyenek, akkor ez az általánosabb út fogalom megengedi, hogy az ilyen utak irányított kört, köröket tartalmazzanak. Az eddig tárgyalt legrövidebb út problémáknál elegendő volt az egyszerű utakra szorítkozni. Valóban, ha megengedtük volna a köröket tartalmazó utakat is, ez nem okozott volna problémát, mivel a tárgyalt esetekben ezen körök hosszai nemnegatívak, így elhagyva az útból az ilyen köröket, az eredmény egy egyszerű út lenne, melynek hossza nem nagyobb, mint a tekintett úté.

A k-adik legrövidebb utak problémájánál nem ilyen egyszerű a helyzet. Tulajdonképpen két különböző problémáról van szó attól függően, hogy út alatt az egyszerű út vagy az általános út fogalmát értjük. Az egyszerű út esetére J. Y. Yen [180] adott egy hatékony megoldó eljárást 1971-ben. Az általános út esetére E. Q. V. Martins [135] adott egy éltörléseken alapuló algoritmust, amelyet később munkatásaival továbbfejlesztettek a [6] dolgozatban. Mi most egy, az általános esetre vonatkozó, D. R. Shiertől származó eljárással fogunk megismerkedni.

Illusztrációként tekintsük a következő példát.

4.4. példa. Tekintsük azt a hálózatot, amelynek csúcsai 1,2,3,4, továbbá élei $(1,2),(1,3),(2,3),(2,4),(3,1),(3,2),(3,4)$. Az élek súlyai az élek felsorolásának sorrendjében: $-1,2,1,4,1,1,1$. Határozzuk meg az 1-ből a 4-be vezető első, második, harmadik legrövidebb úthosszt.

Egyszerűen leellenőrizhető, hogy

az első legrövidebb úthossz 1. (Az $1,2,3,4$ csúcsokat összekötő úthoz tartozik.)

a második legrövidebb úthossz 2. (Az $1,2,3,1,2,3,4$ csúcsokat összekötő úthoz tartozik.)

a harmadik legrövidebb úthossz 3. (Az $(1,3),(3,4)$ élekből álló úthoz tartozik.)

Olyan hálózatokra fogunk általános eljárást megadni, amelyek rendelkeznek az (1) tulajdonsággal, azaz nem tartalmaznak negatív hosszúságú kört. Az ismertetésre kerülő eljárás a szakirodalomban *double-sweep algorithm*

néven ismert és D. R. Shiertől [163], [164] származik. Az eljárás megadásához szükségünk van bizonyos előkészületekre.

Jelölje R^k az olyan k elemű sorozatok halmazát, amelyek komponensei vagy valós számok vagy megegyeznek ∞-nel, továbbá a valós komponensek megelőzik a ∞ komponenseket és szigorúan monoton nőnek. Például $k = 5$ esetén $(-2, 0, 6, \infty, \infty) \in R^k$. Az egyszerűbb szóhasználat kedvéért az R^k halmaz elemeit vektoroknak fogjuk nevezni.

Legyen $\mathbf{a} = (a_1, \ldots, a_k)$, $\mathbf{b} = (b_1, \ldots, b_k) \in R^k$.

Az $\mathbf{a}$ és $\mathbf{b}$ vektorok *általánosított minimalizálásán* azt az $\mathbf{a} \bigoplus \mathbf{b}$-vel jelölt vektort értjük, amelynek komponensei az $\{a_1, \ldots, a_k, b_1, \ldots, b_k\}$ halmaz k legkisebb különböző eleméből állnak és szigorúan monoton nőnek. Például $k = 5$ mellett

$$(2, 3, 4, \infty, \infty) \bigoplus (2, 4, \infty, \infty, \infty) = (2, 3, 4, \infty, \infty).$$

Az $\mathbf{a}$ és $\mathbf{b}$ vektorok *általánosított összeadásán* azt az $\mathbf{a} \biguplus \mathbf{b}$-vel jelölt vektort értjük, amelynek komponensei az $\{a_i + b_j : \ 1 \leq i \leq k; \ 1 \leq j \leq k\}$ halmaz k legkisebb különböző eleméből állnak és szigorúan monoton nőnek. Például $k = 5$ esetén

$$(-1, 4, 8, \infty, \infty) \biguplus (-1, 6, \infty, \infty, \infty) = (-2, 3, 5, 7, 10).$$

Tekintsünk most egy n csúcsból álló hálózatot a c_{ij} élhosszakkal és legyen $1 \leq k \leq n$. Legyen $\mathbf{d}_{ij} = (c_{ij}, \infty, \ldots, \infty)$ minden $1 \leq i, j \leq n$ párosra, ahol a sorozatban a ∞ szimbólum $(k-1)$-szer szerepel.

Most konstruáljuk meg a $\mathbf{D}$ mátrixból az $\mathbf{L}$ és az $\mathbf{U}$ mátrixokat a következők szerint.

(1) Az $\mathbf{L}$-t úgy kapjuk $\mathbf{D}$-ből, hogy $\mathbf{D}$ minden diagonálisban vagy felette levő elemét $(\infty, \infty, \ldots, \infty)$-re cseréljük.

(2) Az $\mathbf{U}$-t úgy kapjuk $\mathbf{D}$-ből, hogy $\mathbf{D}$ minden diagonálisban vagy alatta levő elemét $(\infty, \infty, \ldots, \infty)$-re cseréljük.

Most már készek vagyunk a D. R. Shier [164] által 1976-ban kidolgozott eljárás megadására, amely alkalmas a k-adik legrövidebb úthosszak meghatározására.

Double-Sweep algoritmus ([164])

Előkészítő rész

Legyen $\mathbf{d}_{11}^{(0)} = (0, \infty, \ldots, \infty)$ és $\mathbf{d}_{1,j}^{(0)} = (\infty, \ldots, \infty)$, $(j = 2, \ldots, n)$, $r = 0$.

Iterációs rész (r. iteráció)

- *1. lépés.*

  Határozzuk meg a $\mathbf{d}_{1j}^{(2r+1)}$ vektorokat úgy, hogy az alábbi egyenletek teljesüljenek a $j = n, n-1, \ldots, 1$ indexekre,

  $$\mathbf{d}_{1j}^{(2r+1)} = (\mathbf{d}_{11}^{(2r+1)} \uplus \mathbf{l}_{1j}) \bigoplus (\mathbf{d}_{12}^{(2r+1)} \uplus \mathbf{l}_{2j}) \bigoplus \ldots \bigoplus (\mathbf{d}_{1n}^{(2r+1)} \uplus \mathbf{l}_{nj}) \bigoplus \mathbf{d}_{1j}^{(2r)}$$

  Határozzuk meg a $\mathbf{d}_{1j}^{(2r+2)}$ vektorokat úgy, hogy az alábbi egyenletek teljesüljenek a $j = 1, 2, \ldots, n$ indexekre,

  $$\mathbf{d}_{1j}^{(2r+2)} = (\mathbf{d}_{11}^{(2r+2)} \uplus \mathbf{u}_{1j}) \bigoplus (\mathbf{d}_{12}^{(2r+2)} \uplus \mathbf{u}_{2j}) \bigoplus \ldots \bigoplus (\mathbf{d}_{1n}^{(2r+2)} \uplus \mathbf{u}_{nj}) \bigoplus \mathbf{d}_{1j}^{(2r+1)}$$

- *2. lépés.* (Ellenőrzés.) Ha valamely $t > 1$ indexre $\mathbf{d}_{1j}^{(t)} = \mathbf{d}_{1j}^{(t-1)}$, $(j = 1, \ldots, n)$, akkor vége az eljárásnak, a $\mathbf{d}_{1j}^{(t)}$ vektor komponensei szolgáltatják az 1-ből a j-be vezető legrövidebb úthosszakat, növekvő sorrendben. Ha a feltétel nem teljesül, akkor növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

Az eljárás végrehajtását a 4.4. példán $k = 3$-ra illusztráljuk. A 4.4. példa hálózatának mátrix-reprezentációja a következő:

$$\mathbf{C} = \begin{pmatrix} \infty & -1 & 2 & \infty \\ \infty & \infty & 1 & 4 \\ 1 & 1 & \infty & 1 \\ \infty & \infty & \infty & \infty \end{pmatrix}$$

Ekkor

$$\mathbf{D} = \begin{pmatrix} (\infty,\infty,\infty) & (-1,\infty,\infty) & (2,\infty,\infty) & (\infty,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (1,\infty,\infty) & (4,\infty,\infty) \\ (1,\infty,\infty) & (1,\infty,\infty) & (\infty,\infty,\infty) & (1,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \end{pmatrix}$$

$$\mathbf{L} = \begin{pmatrix} (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \\ (1,\infty,\infty) & (1,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \end{pmatrix}$$

$$\mathbf{U} = \begin{pmatrix} (\infty,\infty,\infty) & (-1,\infty,\infty) & (2,\infty,\infty) & (\infty,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (1,\infty,\infty) & (4,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (1,\infty,\infty) \\ (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) & (\infty,\infty,\infty) \end{pmatrix}$$

Vegyük észre, hogy a definíciókból adódnak a következő azonosságok:

$$(d_1,\dots,d_k)\biguplus(\infty,\dots,\infty)=(\infty,\dots,\infty)$$

$$(d_1,\dots,d_k)\bigoplus(\infty,\dots,\infty)=(d_1,\dots,d_k)$$

Ezeket az azonosságokat és a konkrét $\mathbf{D}$, $\mathbf{L}$, $\mathbf{U}$ márixokat felhasználva egyszerűsíthetjük a $\mathbf{d}_{ij}^{(t)}$ vektorokat definiáló egyenleteket.

Véve az első egyenletet:

$\mathbf{d}_{14}^{(2r+1)}=(\mathbf{d}_{11}^{(2r+1)}\biguplus\mathbf{l}_{14})\bigoplus(\mathbf{d}_{12}^{(2r+1)}\biguplus\mathbf{l}_{24})\bigoplus\ldots\bigoplus(\mathbf{d}_{14}^{(2r+1)}\biguplus\mathbf{l}_{44})\bigoplus\mathbf{d}_{14}^{(2r)}$

Mivel $\mathbf{l}_{t4}=(\infty,\infty,\infty)$, $(t=1,2,3,4)$, ezért az összes zárójelek közti $\biguplus$ művelet eredménye a (∞,∞,∞) vektor.

De akkor $\mathbf{d}_{14}^{(2r+1)}=(\infty,\infty,\infty)\bigoplus\mathbf{d}_{14}^{(2r)}=\mathbf{d}_{14}^{(2r)}$, azaz az első egyenletünk:

(1) $\mathbf{d}_{14}^{(2r+1)}=\mathbf{d}_{14}^{(2r)}$. Véve a második egyenletet

$\mathbf{d}_{13}^{(2r+1)}=(\mathbf{d}_{11}^{(2r+1)}\biguplus\mathbf{l}_{13})\bigoplus(\mathbf{d}_{12}^{(2r+1)}\biguplus\mathbf{l}_{23})\bigoplus\ldots\bigoplus(\mathbf{d}_{14}^{(2r+1)}\biguplus\mathbf{l}_{43})\bigoplus\mathbf{d}_{13}^{(2r)}$

Mivel $\mathbf{l}_{t3}=(\infty,\infty,\infty)$ $(t=1,2,3,4)$, ezért a második egyenletünk:

(2) $\mathbf{d}_{13}^{(2r+1)}=\mathbf{d}_{13}^{(2r)}$ A harmadik egyenletnél

$\mathbf{d}_{12}^{(2r+1)}=(\mathbf{d}_{11}^{(2r+1)}\biguplus\mathbf{l}_{12})\bigoplus(\mathbf{d}_{12}^{(2r+1)}\biguplus\mathbf{l}_{22})\bigoplus\ldots\bigoplus(\mathbf{d}_{14}^{(2r+1)}\biguplus\mathbf{l}_{42})\bigoplus\mathbf{d}_{12}^{(2r)}$

Itt most $\mathbf{l}_{12}=\mathbf{l}_{22}=\mathbf{l}_{42}=(\infty,\infty,\infty)$, továbbá $\mathbf{l}_{32}=(1,\infty,\infty)$. Így a harmadik egyenlet:

(3) $\mathbf{d}_{12}^{(2r+1)}=(\mathbf{d}_{13}^{(2r+1)}\biguplus(1,\infty,\infty))\bigoplus\mathbf{d}_{12}^{(2r)}$

Hasonlóan folytatva az egyenletek egyszerűsítését, az alábbi 8 egyenlethez jutunk:

(1) $\mathbf{d}_{14}^{(2r+1)}=\mathbf{d}_{14}^{(2r)}$

(2) $\mathbf{d}_{13}^{(2r+1)}=\mathbf{d}_{13}^{(2r)}$

(3) $\mathbf{d}_{12}^{(2r+1)}=(\mathbf{d}_{13}^{(2r+1)}\biguplus(1,\infty,\infty))\bigoplus\mathbf{d}_{12}^{(2r)}$

(4) $\mathbf{d}_{11}^{(2r+1)}=(\mathbf{d}_{13}^{(2r+1)}\biguplus(1,\infty,\infty))\bigoplus\mathbf{d}_{11}^{(2r)}$

(5) $\mathbf{d}_{11}^{(2r+2)}=\mathbf{d}_{11}^{(2r+1)}$

(6) $\mathbf{d}_{12}^{(2r+2)}=(\mathbf{d}_{11}^{(2r+2)}\biguplus(-1,\infty,\infty))\bigoplus\mathbf{d}_{12}^{(2r+1)}$

(7) $\mathbf{d}_{13}^{(2r+2)}=(\mathbf{d}_{11}^{(2r+2)}\biguplus(2,\infty,\infty))\bigoplus(\mathbf{d}_{12}^{(2r+2)}\biguplus(1,\infty,\infty))\bigoplus\mathbf{d}_{13}^{(2r+1)}$

(8) $\mathbf{d}_{14}^{(2r+2)}=(\mathbf{d}_{12}^{(2r+2)}\biguplus(4,\infty,\infty))\bigoplus(\mathbf{d}_{13}^{(2r+2)}\biguplus(1,\infty,\infty))\bigoplus\mathbf{d}_{14}^{(2r+1)}$

Ezek után a kapott nyolc szabály alapján ki tudjuk számítani a $\mathbf{d}_{1j}^{(t)}$ vektorokat. A tekintett példára a $\mathbf{d}_{1j}^{(t)}$ vektorok komponenseinek számításait a 4.1. táblázatban foglaltuk össze, ahol a vektorok felett jobbra feltüntettük, hogy az illető vektort mely szabály alkalmazásával számítottuk ki.

t	$\mathbf{d}_{11}^{(t)}$	$\mathbf{d}_{12}^{(t)}$	$\mathbf{d}_{13}^{(t)}$	$\mathbf{d}_{14}^{(t)}$
0	$(0,\infty,\infty)$	(∞,∞,∞)	(∞,∞,∞)	(∞,∞,∞)
1	$(0,\infty,\infty)^{(4)}$	$(\infty,\infty,\infty)^{(3)}$	$(\infty,\infty,\infty)^{(2)}$	$(\infty,\infty,\infty)^{(1)}$
2	$(0,\infty,\infty)^{(5)}$	$(-1,\infty,\infty)^{(6)}$	$(0,2,\infty)^{(7)}$	$(1,3,\infty)^{(8)}$
3	$(0,1,3)^{(4)}$	$(-1,1,3)^{(3)}$	$(0,2,\infty)^{(2)}$	$(1,3,\infty)^{(1)}$
4	$(0,1,3)^{(5)}$	$(-1,0,1)^{(6)}$	$(0,1,2)^{(7)}$	$(1,2,3)^{(8)}$
5	$(0,1,2)^{(4)}$	$(-1,0,1)^{(3)}$	$(0,1,2)^{(2)}$	$(1,2,3)^{(1)}$
6	$(0,1,2)^{(5)}$	$(-1,0,1)^{(6)}$	$(0,1,2)^{(7)}$	$(1,2,3)^{(8)}$

4.1. táblázat. A k-adik legrövidebb úthosszak számítása a 4.4 példára.

Miután az egész fejezetben csak irányított gráfokkal foglalkoztunk, jogosan vethető fel a következő kérdés. Hogyan határozhatók meg irányítatlan gráfokban a legrövidebb utak? A megoldás nagyon egyszerű. Az irányítatlan hálózatból képezünk egy irányított hálózatot úgy, hogy az u, v végpontú e élet helyettesítjük az (u, v) és (v, u) irányított élekkel, és a két élhez súlyként hozzárendeljük az e él súlyát. Egyszerűen belátható, hogy amennyiben $\mathcal{P}$ egy legrövidebb irányított út u-ból v-be az irányított hálózatban, akkor elhagyva az élek irányítását $\mathcal{P}$-ben, az eredmény egy legrövidebb irányítatlan út u-ból v-be a tekintett irányítatlan hálózatban.

A legrövidebb utak elmélete fontos szerepet játszik gyakorlati forgalomszervezési és forgalomtervezési feladatokban. Az alkalmazások iránt érdeklődőknek a teljesség igénye nélkül ajánljuk a [9], [10], [136], [137], [138], [139], [151] tanulmányokat és munkákat.

5. Multiterminális hálózatok

Abban az esetben, ha egy hálózat minden pontpárja között kell a legrövidebb utakat meghatározni, szokásos *multiterminális hálózatról* beszélni, bár a hálózat nem különbözik az előző fejezetben vizsgált hálózatoktól, csak a megoldandó feladat más, a korábbi $n-1$ legrövidebb út helyett most $n(n-1)$ legrövidebb utat kell meghatározni.

Egy megoldási lehetőség lenne, hogy n számú különálló kezdőpontot választanánk, és az előző fejezet valamelyik alkalmas módszerével az így előálló, n különböző legrövidebb út problémákat megoldanánk egymástól függetlenül. Ennél azonban szerencsésebb és hatékonyabb is, ha egyidejűleg kíséreljük meg a legrövidebb utakat meghatározni, és a menet közben nyert információkat minden szükséges legrövidebb út meghatározásához felhasználni. Ezt az elvet valósítja meg az alábbi iterációs eljárás, amely olyan hálózatokra alkalmazható, melyek nem tartalmaznak negatív hosszúságú kört. Nem ismeretes, hogy az algoritmus kitől származik. Az eljárásban az előző fejezetben alkalmazott mátrix-reprezentációt használjuk azzal az eltéréssel, hogy a (d_{ij}) mátrix diagonális elemeit 0-ra változtatjuk, azaz az új reprezentációban $d_{ii} = 0$, $i = 1, \ldots, n$.

Multiterminális eljárás

Előkészítő rész

Legyen $u_{ii}^{(0)} = 0$, $i = 1, \ldots, n$, és $u_{ij}^{(0)} = +\infty$ minden $i \neq j \in \{1, \ldots, n\}$ számpárra. Továbbá adjuk r-nek a 0 értéket.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $r = n - 1$, akkor vége az eljárásnak, $u_{ij}^{(n-1)}$ az i csúcsból a j csúcsba vezető, legrövidebb út hosszát adja meg minden $i \neq j \in \{1, \ldots, n\}$ csúcspárra. Ha $r < n - 1$, akkor térjünk rá a 2. lépésre.

- *2. lépés.* Minden $i, j \in \{1, \ldots, n\}$ csúcspárra legyen

$$u_{ij}^{(r+1)} = \min_{1 \leq k \leq n} \{u_{ik}^{(r)} + d_{kj}\}.$$

  Növeljük r értékét 1-gyel és térjünk rá a következő iterációra.

Az eljárás helyességének igazolásához m szerinti teljes indukcióval bizonyítjuk a következő állítást:

(i) Minden $i \neq j \in \{1, \ldots, n\}$ csúcspárra $u_{ij}^{(m)}$ az i csúcsból a j csúcsba vezető, legfeljebb m élet tartalmazó legrövidebb út hossza, továbbá $u_{ii}^{(m)} = 0$, $i = 1, \ldots, n$.

Ha $m = 0$, akkor az (i) állítás nyilvánvalóan teljesül. Ezek után legyen $0 \leq m < n - 1$ tetszőleges, és tegyük fel, hogy az (i) állítás teljesül m-re.

Elsőként vizsgáljuk $u_{ii}^{(m+1)}$ értékét egy tetszőleges i csúcsra. Vegyük észre, hogy $u_{ii}^{(m+1)}$ definíciójában k felveheti az i értéket is. Ekkor a megfelelő kifejezés $u_{ii}^{(m)} + d_{ii}$, ami az indukciós feltevés és d_{ii} megváltoztatott definíciója miatt 0-val egyenlő. Másrészt bármely i-től különböző k csúcsra $u_{ik}^{(m)} + d_{ki}$ nemnegatív. Valóban, ellenkező esetben az indukciós feltevés szerint $u_{ik}^{(m)}$ az i-ből a k-ba vezető, legfeljebb m élet tartalmazó legrövidebb út hosszát adja meg, és azt kapnánk, hogy meghosszabbítva egy ilyen utat a (k, i) éllel, egy negatív hosszúságú kör állna elő, ami ellentmond a feltevésünknek, tehát $u_{ik}^{(m)} + d_{ki} \geq 0$ minden k-ra. Ebből már következik, hogy a tekintett minimum értéke 0, azaz $u_{ii}^{(m+1)} = 0$.

Most tekintsünk egy tetszőleges $i \neq j$ csúcspárt és vegyünk egy i-ből j-be vezető legrövidebb utat, amely legfeljebb $m+1$ élet tartalmaz. Ilyen út biztosan lesz, legfeljebb tartalmaz nemlétező éleket. Legyen ebben az útban j őse k. Akkor, mivel az utak páronként különböző csúcsokat tartalmaznak, ezért $k \neq j$. Másrészt az indukciós feltevés szerint az i csúcsból a k-ba vezető, legfeljebb m élet tartalmazó legrövidebb út hossza $u_{ik}^{(m)}$. (Megjegyezzük, hogy ez $i = k$ esetén is igaz, ekkor $u_{ii}^{(m)} = 0$.) A tekintett útból elhagyva az utolsó élet, egy i-ből k-ba vezető, legfeljebb m élet tartalmazó utat kapunk, így a tekintett út hossza nem kisebb, mint $u_{ik}^{(m)} + d_{kj}$. Mivel a tekintett út egy legrövidebb út és legfeljebb $m+1$ élet tartalmaz, ezért hossza nem lehet nagyobb, mint $u_{ik}^{(m)} + d_{kj}$. Következésképpen, a tekintett út hossza pontosan $u_{ik}^{(m)} + d_{kj}$. Ezek után legyen $s \neq k$, $s \in \{1, \ldots, n\} \setminus \{j\}$, és vegyünk egy tetszőleges i-ből j-be vezető olyan utat, amelynek utolsó éle (s, j). Akkor az indukciós feltevés alapján a legfeljebb $m + 1$ élet tartalmazó ilyen utak közül a legrövidebbek hossza $u_{is}^{(m)} + d_{sj}$. De a korábban tekintett út az i csúcsból a j csúcsba vezető, legfeljebb $m + 1$ élet tartalmazó legrövidebb út volt, így hossza nem nagyobb, mint $u_{is}^{(m)} + d_{sj}$. Ezzel azt kaptuk, hogy minden $s \in \{1, \ldots, n\} \setminus \{j\}$ csúcsra

$$u_{ik}^{(m)} + d_{kj} \leq u_{is}^{(m)} + d_{sj}.$$

Vizsgáljuk most a minimumban szereplő kifejezést $s = j$-re. Ebben az esetben a kifejezés $u_{ij}^{(m)}$-vel egyenlő. De akkor ez az indukciós feltevés alapján egy, az i csúcsból a j csúcsba vezető legfeljebb m élet tartalmazó legrövidebb út hossza, ami nem lehet kisebb, mint $u_{ik}^{(m)} + d_{kj}$ mivel a tekintett utunk legfeljebb $m+1$ élet tartalmazó, i-ből j-be vezető legrövidebb út volt. Tehát

$$u_{ik}^{(m)} + d_{kj} = \min_{1 \leq k \leq n} \{u_{ik}^{(m)} + d_{kj}\},$$

amivel az (i) állítást igazoltuk.

Az eljárás műveletigényét illetően vegyük észre, hogy az eljárás során az $\mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_{n-1}$ $n \times n$-es mátrixok elemeit kell kiszámítani, ami összesen n^3 elem kiszámítását jelenti. Másrészt minden egyes elem kiszámításához n összeadás és n összehasonlítás szükséges. Így az eljárás műveletigénye $O(n^4)$. Ez pontosan azt jelenti, hogy a multiterminális eljárás a műveletigényt illetően érdemben nem különbözik egy olyan eljárástól, amelyben a fokozatos közelítés módszerét hajtanánk végre n-szer egymástól függetlenül. Tehát úgy tűnik, hogy az összes legrövidebb úthossz egyidejű számítása nem eredményez hatékonyságnövekedést. Szerencsére nem ennyire rossz a helyzet, amint azt az alábbiakban látni fogjuk.

A multiterminális eljárás hatékonyságának javítása az eljárás egy más végrehajtási technikáján alapul. Ehhez vegyük észre, hogy az $\mathbf{U}_{m+1}$ mátrix $u_{ij}^{(m+1)}$ elemének kiszámításához szükséges minimumképzés művelete nagyon hasonlít egy szorzatmátrix (i, j) indexű elemének kiszámításához. Valóban, tetszőleges $\mathbf{A} = (a_{ij})$ és $\mathbf{B} = (b_{ij})$ valós mátrixokra definiáljuk a $\otimes$ műveletet a következők szerint:

$$\mathbf{A} \otimes \mathbf{B} = \mathbf{C},$$

ahol $c_{ij} = \min_{1 \leq k \leq n} \{a_{ik} + b_{kj}\}$. A definiált műveletről egyszerűen belátható, hogy asszociatív az $n \times n$-es valós mátrixok halmazán. Másrészt az eljárásból adódik, hogy

$$\mathbf{U}_0 \otimes \mathbf{D} = \mathbf{D},$$

$$\mathbf{U}_1 = \mathbf{U}_0 \otimes \mathbf{D} = \mathbf{D},$$

$$\mathbf{U}_2 = \mathbf{U}_1 \otimes \mathbf{D} = \mathbf{D} \otimes \mathbf{D},$$

$$\mathbf{U}_3 = \mathbf{U}_2 \otimes \mathbf{D} = \mathbf{D} \otimes \mathbf{D} \otimes \mathbf{D},$$

és hasonlóan folytatva a sorozatot, $\mathbf{U}_{n-1}$-re azt kapjuk, hogy

$$\mathbf{U}_{n-1} = \mathbf{U}_{n-2} \otimes \mathbf{D} = \mathbf{D} \otimes \ldots \otimes \mathbf{D},$$

ahol az utolsó kifejezésben a tényezők száma $n-1$. Ez az asszociativitás miatt pontosan azt jelenti, hogy minden $1 \leq j \leq n-1$ indexre $\mathbf{U}_j = \mathbf{D}^j$, ahol $\mathbf{D}^j$ a $\mathbf{D}$ mátrix j-edik hatványát jelöli a $\otimes$ műveletre vonatkozóan.

Mivel $\mathbf{U}_{n-1}$ a legfeljebb $n-1$ élet tartalmazó, legrövidebb utak hosszait adja meg, és minden út legfeljebb $n-1$ élet tartalmazhat egy n szögpontú gráfban, ezért egyszerűen belátható, hogy $\mathbf{U}_{n-1} = \mathbf{U}_t$ minden $t \geq n-1$ pozitív egészre. Most válasszunk egy olyan t pozitív egész számot, amelyre $2^t \geq n-1$ teljesül. Akkor tudjuk képezni a

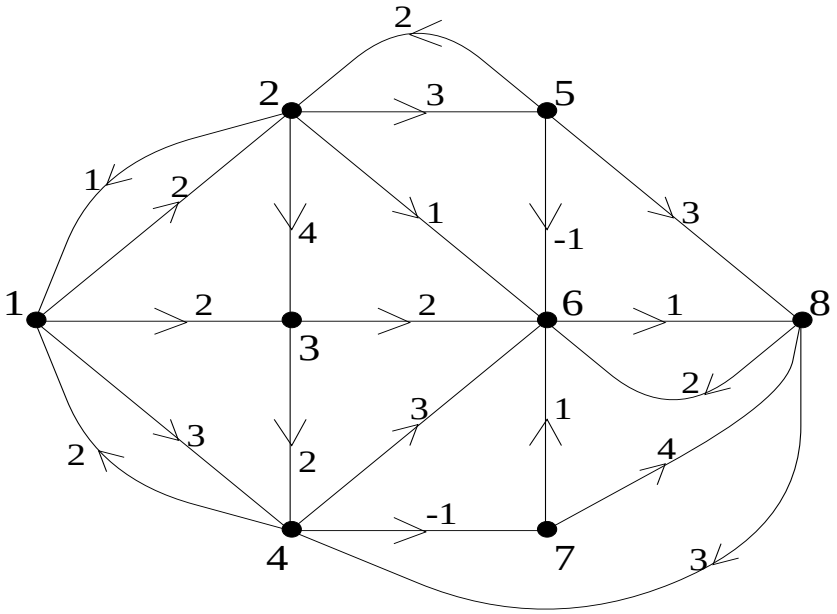
$$\mathbf{D},\ \mathbf{D}^2,\ \mathbf{D}^{2^2},\ \ldots, \mathbf{D}^{2^t}$$

mátrixsorozatot úgy, hogy a sorozat minden tagját úgy számítjuk ki, hogy a megelőző tagot szorozzuk önmagával a $\otimes$ művelettel. Mivel $2^t \geq n-1$, ezért $\mathbf{D}^{2^t} = \mathbf{U}_{n-1}$. Másrészt a fenti mátrixsorozat előállításához t számú mátrixszorzást kell végrehajtanunk a $\otimes$ szorzási művelettel. Egy ilyen szorzás végrehajtásának a műveletigénye $O(n^3)$, és így az egész eljárás műveletigénye a fentiekben vázolt végrehajtás mellett $O(n^3 \log_2(n-1))$.

Az eljárás demonstrálására vegyük a következő példát.

5.1. példa. Tekintsük azon hálózatot, melynek csúcspontjai $1, \ldots, 8$, és az éleket valamint azok hosszait az alábbi $\mathbf{D}$ mátrix adja meg. A módosított reprezentációnak megfelelően a diagonális elemek rendre 0-val egyenlőek az alábbi mátrixban. A nemlétező élek esetében, ahogy már korábban is, a $+\infty$-t egyszerűen ∞-nel helyettesítettük és ezt a jelölést fogjuk használni az eljárás során is. A hálózat grafikus reprezentációját mutatja a 5.1. ábra.

$$\begin{array}{cccccccc}
0 & 2 & 2 & 3 & \infty & \infty & \infty & \infty \\
1 & 0 & 4 & \infty & 3 & 1 & \infty & \infty \\
\infty & \infty & 0 & 2 & \infty & 2 & \infty & \infty \\
2 & \infty & \infty & 0 & \infty & 3 & -1 & \infty \\
\infty & 2 & \infty & \infty & 0 & -1 & \infty & 3 \\
\infty & \infty & \infty & \infty & \infty & 0 & \infty & 1 \\
\infty & \infty & \infty & \infty & \infty & 1 & 0 & 4 \\
\infty & \infty & \infty & 3 & \infty & 2 & \infty & 0
\end{array}$$



5.1. ábra. Az 5.1. példa hálózatának grafikus reprezentációja.

Mivel $2^3 = 8 \geq n - 1 = 7$, ezért a $\mathbf{D} \otimes \mathbf{D} = \mathbf{D}^2$, $\mathbf{D}^2 \otimes \mathbf{D}^2 = \mathbf{D}^4$ és $\mathbf{D}^4 \otimes \mathbf{D}^4 = \mathbf{D}^8$ három $\otimes$ szorzási műveletet kell végrehajtanunk. A műveletek végrehajtása során előálló mátrixok rendre a következők:

$$\mathbf{D}^2 = \begin{pmatrix} 0 & 2 & 2 & 3 & 5 & 3 & 2 & \infty \\ 1 & 0 & 3 & 4 & 3 & 1 & \infty & 2 \\ 4 & \infty & 0 & 2 & \infty & 2 & 1 & 3 \\ 2 & 4 & 4 & 0 & \infty & 0 & -1 & 3 \\ 3 & 2 & 6 & 6 & 0 & -1 & \infty & 0 \\ \infty & \infty & \infty & 4 & \infty & 0 & \infty & 1 \\ \infty & \infty & \infty & 7 & \infty & 1 & 0 & 2 \\ 5 & \infty & \infty & 3 & \infty & 2 & 2 & 0 \end{pmatrix}$$

$$\mathbf{D}^4 = \begin{pmatrix} 0 & 2 & 2 & 3 & 5 & 3 & 2 & 4 \\ 1 & 0 & 3 & 4 & 3 & 1 & 3 & 2 \\ 4 & 6 & 0 & 2 & 9 & 2 & 1 & 3 \\ 2 & 4 & 4 & 0 & 7 & 0 & -1 & 1 \\ 3 & 2 & 5 & 3 & 0 & -1 & 2 & 0 \\ 6 & 8 & 8 & 4 & \infty & 0 & 3 & 1 \\ 7 & 11 & 11 & 5 & \infty & 1 & 0 & 2 \\ 5 & 7 & 7 & 3 & 10 & 2 & 2 & 0 \end{pmatrix}$$

$$\mathbf{D}^8 = \begin{pmatrix} 0 & 2 & 2 & 3 & 5 & 3 & 2 & 4 \\ 1 & 0 & 3 & 4 & 3 & 1 & 3 & 2 \\ 4 & 6 & 0 & 2 & 9 & 2 & 1 & 3 \\ 2 & 4 & 4 & 0 & 7 & 0 & -1 & 1 \\ 3 & 2 & 5 & 3 & 0 & -1 & 2 & 0 \\ 6 & 8 & 8 & 4 & 11 & 0 & 3 & 1 \\ 7 & 9 & 9 & 5 & 12 & 1 & 0 & 2 \\ 5 & 7 & 7 & 3 & 10 & 2 & 2 & 0 \end{pmatrix}$$

A kitűzött probléma megoldására ismeretes egy hatékonyabb eljárás is, amely R. W. Floyd-tól származik [56] és alapvetően S. Warshall [174] dolgozatának egy tételére épül. Ennek következtében az eljárásra több szerző is Floyd-Warshall módszer néven hivatkozik; mi is ezt az elnevezést fogjuk használni. Az eljárás szintén olyan hálózatokra alkalmazható, amelyek nem tartalmaznak negatív hosszúságú kört.

Floyd-Warshall módszer ([56])

Előkészítő rész
Legyen $u_{ii}^{(0)} = 0$, $i = 1, \ldots, n$, és $u_{ij}^{(0)} = d_{ij}$ minden $i \neq j \in \{1, \ldots, n\}$ számpárra. Továbbá adjuk r-nek a 0 értéket.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $r = n$, akkor vége az eljárásnak, $u_{ij}^{(n)}$ az i csúcsból a j csúcsba vezető, legrövidebb út hosszát adja meg minden $i \neq j \in \{1, \ldots, n\}$ csúcspárra. Ha $r < n$, akkor térjünk rá a 2. lépésre.

- *2. lépés.* Minden $i, j \in \{1, \ldots, n\}$ csúcspárra legyen

$$u_{ij}^{(r+1)} = \min\{u_{i,r+1}^{(r)} + u_{r+1,j}^{(r)}, u_{ij}^{(r)}\}.$$

  Növeljük r értékét 1-gyel és térjünk rá a következő iterációra.

Az eljárás helyességének igazolásához m szerinti teljes indukcióval megmutatjuk, hogy

(a) minden $i \neq j \in \{1, \ldots, n\}$ csúcspárra $u_{ij}^{(m)}$ az i csúcsból a j csúcsba vezető, legfeljebb az $1, \ldots, m$ közbülső csúcsokat (nem kezdőpontja és nem végpontja az útnak) tartalmazó legrövidebb út hossza, továbbá $u_{ii}^{(m)} = 0$, $i = 1, \ldots, n$.

Az $m = 0$ esetben $u_{ii}^{(0)} = 0$, $(i = 1, \ldots, n)$ definíció szerint, továbbá $u_{ij}^{(0)} = d_{ij}$ az i-ből j-be vezető, közbülső csúcsot nem tartalmazó legrövidebb út hossza. Tehát $m = 0$-ra teljesül az (a) állítás.

Most legyen $0 \leq m < n$ és tegyük fel, hogy az m egészre teljesül az (a) állítás. Megmutatjuk, hogy ekkor (a) $m + 1$-re is érvényes.

Legyen $1 \leq i \leq n$ és vizsgáljuk elsőként az $u_{ii}^{(m+1)}$ értéket. Definíció szerint

$$u_{ii}^{(m+1)} = \min\{u_{i,m+1}^{(m)} + u_{m+1,i}^{(m)}, u_{ii}^{(m)}\}.$$

Vegyük észre, hogy az indukciós feltevés szerint $u_{ii}^{(m)} = 0$, és így $u_{ii}^{(m+1)} \leq 0$. Másrészt, ha $u_{ii}^{(m+1)} < 0$ lenne, akkor a minimumban szereplő első két tag összegével lenne egyenlő. Ezek közül az első tag egy i-ből $m + 1$-be vezető út hossza, a második tag pedig egy, az $m+1$ csúcsból az i csúcsba vezető út hossza, így a két hossz összege egy körút hossza, ami feltételezésünk alapján nem lehet negatív. Következésképpen, $u_{ii}^{(m+1)} = 0$, $(i = 1, \ldots, n)$.

Most legyen $i \neq j \in \{1, \ldots, n\}$. Megmutatjuk, hogy $u_{ij}^{(m+1)}$ egy, az i csúcsból a j csúcsba vezető, legfeljebb az $1, \ldots, m + 1$ közbülső csúcsokat tartalmazó legrövidebb út hossza. Ehhez tekintsünk egy i-ből j-be vezető, legfeljebb az $1, \ldots, m + 1$ közbülső csúcsokat tartalmazó $\mathcal{P}_{ij}$ legrövidebb utat, melynek a hossza legyen w. Vizsgáljuk most $u_{ij}^{(m+1)}$ defnícióját.

$$u_{ij}^{(m+1)} = \min\{u_{i,m+1}^{(m)} + u_{m+1,j}^{(m)}, u_{ij}^{(m)}\}$$

Az indukciós feltevés szerint $u_{i,m+1}^{(m)}$ egy olyan út hossza, amely i-ből $m + 1$-be vezet és legfeljebb az $1, \ldots, m$ közbülső csúcsokat tartalmazza. Hasonlóan $u_{m+1,j}^{(m)}$ egy $m+1$-ből j-be vezető, legfeljebb az $1, \ldots, m$ közbülső csúcsokat tartalmazó út hossza. Összekapcsolva ezt a két utat, egy olyan utat kapunk, amely i-ből j-be vezet, és legfeljebb az $1, \ldots, m + 1$ közbülső csúcsokat tartalmazza. Mivel $\mathcal{P}_{ij}$ is ilyen típusú út és ráadásul legrövidebb, ezért $w \leq u_{i,m+1}^{(m)} + u_{m+1,j}^{(m)}$. Hasonló meggondolással azt kapjuk, hogy $w \leq u_{ij}^{(m)}$. A két egyenlőtlenségből

$$w \leq u_{ij}^{(m+1)}$$

következik. Most megmutatjuk, hogy az egyenlőségnek kell teljesülni. Ehhez különböztessük meg a következő két esetet.

(1) A $\mathcal{P}_{ij}$ út nem tartalmazza az $m+1$ csúcsot. Akkor az indukciós feltevés szerint $u_{ij}^{(m)} \leq w$, amiből $u_{ij}^{(m+1)} \leq w$ következik.

(2) A $\mathcal{P}_{ij}$ út tartalmazza közbülső pontként az $m+1$ csúcsot. Akkor az $m+1$ csúcsnál $\mathcal{P}_{ij}$ szétvágható egy i-ből $m+1$-be vezető $\mathcal{P}_{i,m+1}$ és egy $m+1$-ből j-be vezető $\mathcal{P}_{m+1,j}$ útra és w a két részút hosszainak az összege. Vegyük észre, hogy a két új út mindegyike közbülső csúcspontként legfeljebb az $1, \ldots, m$ csúcsokat tartalmazza. Akkor viszont az indukciós feltevés alapján $\mathcal{P}_{i,m+1}$ hossza nem kisebb, mint $u_{i,m+1}^{(m)}$ és $\mathcal{P}_{m+1,j}$ hossza nem kisebb, mint $u_{m+1,j}^{(m)}$. Következésképpen, $u_{i,m+1}^{(m)} + u_{m+1,j}^{(m)} \leq w$, amiből ismét $u_{ij}^{(m+1)} \leq w$ következik.

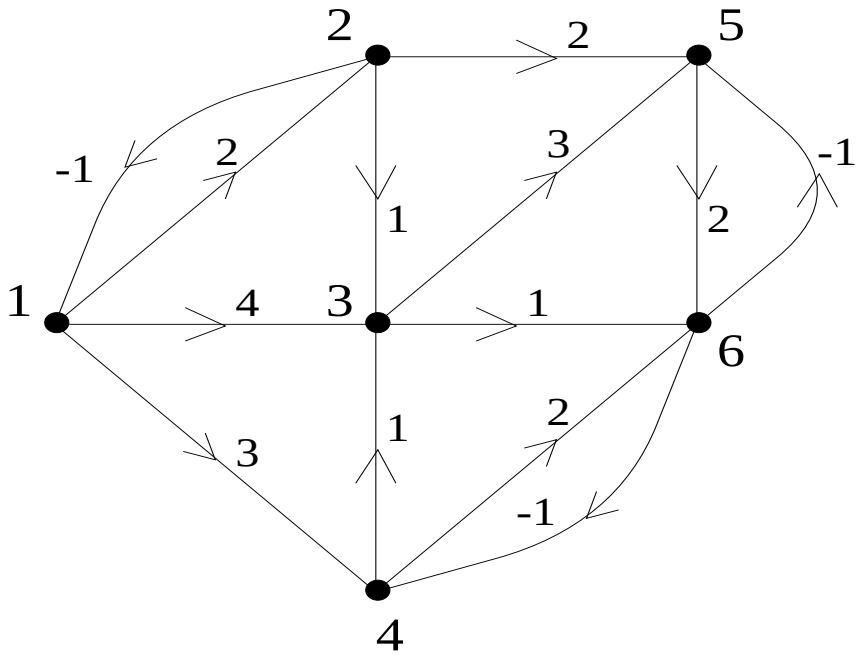
Ezzel az elárás helyességének igazolását befejeztük.

Az eljárás demonstrálására megoldjuk a következő feladatot:

5.2. példa. Tekintsük azt a hálózatot, amelynek csúcspontjai az $1, \ldots, 6$ természetes számok, továbbá éleit és azok hosszait az alábbi mátrix-reprezentáció tartalmazza. Itt is a módosított reprezentációt használjuk, azaz a diagonális elemek rendre 0-val egyenlőek. A nemlétező élek esetében a $+\infty$-t egyszerűen ∞-nel helyettesítettük, és ezt a jelölést fogjuk használni az eljárás végrehajtása során is. A hálózat grafikus reprezentációját mutatja az 5.2. ábra.

$$\mathbf{D} = \begin{pmatrix} 0 & 2 & 4 & 3 & \infty & \infty \\ -1 & 0 & 1 & \infty & 2 & \infty \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & \infty & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & \infty & -1 & -1 & 0 \end{pmatrix}$$

A definíciókból nyilvánvalóan következik, hogy $\mathbf{U}_0 = \mathbf{D}$, így $\mathbf{U}_0$-t nem kell külön kiszámítani. Elvégezve a szükséges számításokat, az $\mathbf{U}_1$ és $\mathbf{U}_2$ mátrixok elemeire a következő értékeket kapjuk:



5.2. ábra. Az 5.2. példa hálózatának grafikus reprezentációja.

$\mathbf{U}_1$

$$\begin{matrix} 0 & 2 & 4 & 3 & \infty & \infty \\ -1 & 0 & 1 & 2 & 2 & \infty \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & \infty & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & \infty & -1 & -1 & 0 \end{matrix}$$

$\mathbf{U}_2$

$$\begin{matrix} 0 & 2 & 3 & 3 & 4 & \infty \\ -1 & 0 & 1 & 2 & 2 & \infty \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & \infty & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & \infty & -1 & -1 & 0 \end{matrix}$$

Az $\mathbf{U}_3$ és $\mathbf{U}_4$ mátrixok elemeire a következő értékek adódnak:

$\mathbf{U}_3$

$$\begin{matrix} 0 & 2 & 3 & 3 & 4 & 4 \\ -1 & 0 & 1 & 2 & 2 & 2 \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & 4 & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & \infty & -1 & -1 & 0 \end{matrix}$$

$\mathbf{U}_4$

$$\begin{matrix} 0 & 2 & 3 & 3 & 4 & 4 \\ -1 & 0 & 1 & 2 & 2 & 2 \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & 4 & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & 0 & -1 & -1 & 0 \end{matrix}$$

Végül az $\mathbf{U}_5$ és $\mathbf{U}_6$ mátrixok elemeire az alábbi értéket nyerjük:

$$\mathbf{U}_5 = \begin{pmatrix} 0 & 2 & 3 & 3 & 4 & 4 \\ -1 & 0 & 1 & 2 & 2 & 2 \\ \infty & \infty & 0 & \infty & 3 & 1 \\ \infty & \infty & 1 & 0 & 4 & 2 \\ \infty & \infty & \infty & \infty & 0 & 2 \\ \infty & \infty & 0 & -1 & -1 & 0 \end{pmatrix} \qquad \mathbf{U}_6 = \begin{pmatrix} 0 & 2 & 3 & 3 & 3 & 4 \\ -1 & 0 & 1 & 1 & 1 & 2 \\ \infty & \infty & 0 & 0 & 0 & 1 \\ \infty & \infty & 1 & 0 & 1 & 2 \\ \infty & \infty & 2 & 1 & 0 & 2 \\ \infty & \infty & 0 & -1 & -1 & 0 \end{pmatrix}$$

Az eljárás műveletigényét illetően vegyük észre, hogy az eljárás során az $\mathbf{U}_1, \ldots, \mathbf{U}_n$ mátrixok elemeit kell kiszámítani, ami pontosan n^3 elem kiszámítását jelenti. Másrészt minden egyes elem kiszámításához egy összeadás és egy összehasonlítás szükséges. Következésképpen, a Floyd-Warshall módszer műveletigénye $O(n^3)$.

A multiterminális hálózatoknál is számíthatók a k-adik legrövidebb úthosszak. Például a Floyd-Warshall módszernek létezik olyan kiterjesztése (ld. [54]), amely alkalmas a k-adik legrövidebb úthosszak számítására.

Ezzel le is zárjuk a hálózati legrövidebb utak meghatározása témájának tárgyalását, azzal a kiegészítéssel, hogy a téma iránt érdeklődőknek szíves figyelmébe ajánljuk az [54] és [122] könyveket, melyek többek között ezt a témakört is szisztematikusan, az itt tárgyaltaknál részletesebben vizsgálják.

6. Hálózati folyamok

A jelen fejezetben olyan speciális irányított hálózatokkal foglalkozunk, amelyekben az élekhez rendelt értékek rendre nemnegatívak, továbbá az illető hálózatokban van egy forrás és egy nyelő. Ennek megfelelően ebben a részben gráfon mindig irányított gráfot értünk.

Egy ilyen hálózat úgy interpretálható, mint egy csőrendszer, amelyben csak az éleknek megfelelő irányban történhet áramlás, és az élhez rendelt valós szám a tekintett él átbocsájtóképessége (pl. egységnyi idő alatt a feltételezett anyagból mennyi tud az élen áthaladni). Az anyag áramlását stacionáriusnak tételezzük fel. Ezek után az a kérdés, hogy mennyi az egész hálózat maximális átbocsájtóképessége.

A fentiekben vázolt feladathoz a következő optimumszámítási modell konstruálható.

Jelölje $\mathcal{G} = (\{1, \ldots, n\}, E)$ a hálózat gráfját, legyen az 1 csúcs a forrás és az n csúcs a nyelő. Jelölje továbbá c_{ij} az (i, j) él átbocsájtóképességét, ha $(i, j) \in E$, ha pedig $(i, j) \notin E$, akkor legyen $c_{ij} = 0$. Végül legyen x_{ij} az (i, j) élen egységnyi idő alatt áthaladó anyagmennyiség, ahol $i \neq j \in \{1, \ldots, n\}$. Akkor a tekintett probléma az alábbi lineáris programozási feladattal írható le:

$$\sum_{(1,t)\in E} x_{1t} = \sum_{(t,n)\in E} x_{tn}$$

$$0 \leq x_{ij} \leq c_{ij},\ i \neq j \in \{1, \ldots, n\} \tag{6.1}$$

$$\sum_{(t,i)\in E} x_{ti} = \sum_{(i,t)\in E} x_{it},\ i \in \{2, \ldots, n-1\}$$

$$\sum_{(1,t)\in E} x_{1t} = z(\mathbf{X}) \to \max$$

A (6.1) feladatot szokásos *folyamproblémának* nevezni. A feltételeknek az alábbi vizuális jelentést lehet adni. Az első feltétel azt írja elő, hogy a

forráson kiáramló anyagmennyiségnek meg kell egyeznie a nyelőn eltávozó anyagmennyiséggel, azaz minden anyag csak a forráson keresztül kerülhet a rendszerbe és csak a nyelőn keresztül távozhat a rendszerből, ráadásul a rendszeren történő áthaladás során anyag nem vész el. A második feltételcsoport azt garantálja, hogy minden élen legfeljebb annyi anyag áramlik át, amennyit az él átbocsájtóképessége lehetővé tesz. Végül az utolsó feltételcsoport azt írja elő, hogy minden, a forrástól és a nyelőtől különböző i csúcsra, az i-be befolyó anyagmennyiségnek meg kell egyeznie az i-ből kifolyó anyagmennyiséggel.

A (6.1) feladat egy $\bar{\mathbf{X}}$ lehetséges megoldását *lehetséges folyamnak* vagy egyszerűen *folyamnak* nevezzük. Vegyük észre, hogy a folyam több, mint az átbocsájtott anyagmennyiség, egyidejűleg tartalmazza azt is, hogy a folyam miként halad át a hálózaton. Az adott folyamhoz tartozó anyagmennyiséget a *folyam értékének* nevezzük. Ez más szóval a tekintett lehetséges megoldáshoz tartozó célfüggvényérték.

Megvizsgálva a (6.1) feladatot, rögtön adódik, hogy annak mindig létezik optimális megoldása. Valóban, ismeretes az, hogy a lineáris programozási feladatok lehetséges megoldásainak a halmaza zárt, másrészt a tekintett feladatnál a második feltételcsoport miatt ez a halmaz korlátos. Nyilvánvalóan a 0-mátrix is lehetséges megoldás. Tehát a lehetséges megoldások L halmaza korlátos zárt nemüres halmaz. Másrészt a célfüggvény lineáris lévén, folytonos L-en. Az elmondottakból már következik, hogy a (6.1) feladatnak létezik optimális megoldása.

Annak ellenére, hogy a fentiek szerint a (6.1) folyamproblémának mindig létezik optimális megoldása, érdemes még ezt a kérdést kicsit vizsgálni, ugyanis bizonyos esetekben csak a 0-mátrix lesz lehetséges megoldás, ami valójában csak annyi információt nyújt, hogy a hálózaton nem lehet átbocsájtani semmit. Ezen vizsgálathoz szükségünk lesz a következő definíciókra.

A $\mathcal{G} = (\{1, \ldots, n\}, E)$ gráf egy olyan $i_1, \ldots, , i_k$ páronként különböző csúcsokból álló csúcssorozatát, amelyre minden $1 \leq t < k$ indexre $(i_t, i_{t+1}) \in E$ vagy $(i_{t+1}, i_t) \in E$ teljesül *irányítatlan útnak* nevezzük. Tehát itt csak az a kikötés, hogy a tekintett csúcsokat élek kössék össze, melyek irányításától most eltekintünk. A $\mathcal{G}$ irányított gráfot *összefüggőnek* nevezzük, ha minden $i, j \in \{1, \ldots, n\}$ csúcspárra létezik $\mathcal{G}$-ben ezen csúcsokat összekötő irányítatlan út. A $\mathcal{G}$ gráf egy maximális összefüggő részgráfját $\mathcal{G}$ *komponensének* nevezzük. A definíciókból egyszerűen adódik, hogy bármely gráf előáll komponenseinek diszjunkt uniójaként.

Térjünk vissza most a folyamprobléma vizsgálatára, melynek gráfját

jelölje $\mathcal{G}$. Ha a $\mathcal{G}$ gráf több komponensből áll, akkor a folyamproblémának csak akkor lehet a triviális, 0-mátrix megoldástól különböző megoldása, ha a forrás és a nyelő egy komponensben vannak, és ebben az esetben a többi komponensnek nincs szerepe a tekintett folyamproblémában. Valóban, a komponensek között nem léteznek élek, így nem lehet anyagáramlás közöttük. Következésképpen, elegendő csak olyan folyamproblémák vizsgálatára szorítkozni, amelyeknek a gráfja összefüggő. Ezt a következőkben minden hivatkozás nélkül feltételezzük.

A továbbiakban azt vizsgáljuk, hogy miként lehet gráfelméleti eszközöket felhasználni a (6.1) feladat megoldásában. Az ismertetésre kerülő eljáráshoz szükségünk lesz a következő definícióra. Legyen $\mathcal{P}$ egy, a forrásból a nyelőbe vezető irányítatlan út a tekintett hálózatban, azaz olyan $i_1, \ldots, i_k$ csúcssorozat, hogy $i_1 = 1$, $i_k = n$ és minden $1 \leq t < k$ indexre $(i_t, i_{t+1}) \in E$ vagy $(i_{t+1}, i_t) \in E$ teljesül. $\mathcal{P}$ valamely (i, j) élét *előremenő élnek* nevezzük, ha a forrástól a nyelő fele van irányítva, ellenkező irányítás esetén pedig *hátramenő élről* beszélünk. Továbbá azt mondjuk, hogy a $\mathcal{P}$ út egy $\bar{\mathbf{X}}$ folyamra nézve *növelő út*, ha $\mathcal{P}$ valamennyi előremenő (i, j) élére $\bar{x}_{ij} < c_{ij}$ és $\mathcal{P}$ valamennyi hátramenő (r, s) élére $0 < \bar{x}_{rs}$ teljesül.

A bevezetett definíció alkalmasnak látszik arra, hogy abban az esetben, ha egy adott $\bar{X}$ folyamhoz létezik egy $\mathcal{P}$ növelő út, akkor $\mathcal{P}$ mentén megváltoztatva az $\bar{x}_{ij}$ értékeket, növeljük a folyam értékét. Ez valóban így van, amint azt a következő tétel mutatja.

6.1. tétel. (Növelő utak tétele.) *Ha az* $\bar{\mathbf{X}} = (\bar{x}_{ij})$ *folyamhoz létezik egy növelő út, akkor megadható egy olyan új* $\mathbf{X}^*$ *folyam, amelynek nagyobb az értéke, mint az* $\bar{\mathbf{X}}$ *folyam értéke.*

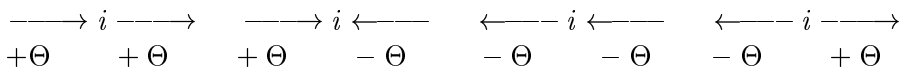
Bizonyítás. Tegyük fel, hogy a $\mathcal{P}$ irányítatlan út növelő út az $\bar{\mathbf{X}}$ folyamra nézve. Jelölje a $\mathcal{P}$ útban szereplő csúcsokat $1 = i_1, i_2, \ldots, i_k = n$ és a $\mathcal{P}$ éleiből álló halmazt Γ. Bontsuk szét a Γ halmazt két diszjunkt részhalmazra; tartalmazza Γ_1 a $\mathcal{P}$ út előremenő éleit, míg Γ_2 a $\mathcal{P}$ út hátramenő éleit. Képezzük a következő minimumokat:

$$\Theta_1 = \min\{c_{ij} - \bar{x}_{ij} : (i, j) \in \Gamma_1\}, \qquad \Theta_2 = \min\{\bar{x}_{ij} : (i, j) \in \Gamma_2\},$$

és legyen $\Theta = \min\{\Theta_1, \Theta_2\}$. Mivel $\mathcal{P}$ növelő út, ezért $\Theta > 0$. Ezek után definiáljuk az $\mathbf{X}^*$ mátrixot a következők szerint:

$$x^*_{ij} = \begin{cases} \bar{x}_{ij}, & \text{ha } (i, j) \notin \Gamma\ , \\ \bar{x}_{ij} + \Theta, & \text{ha } (i, j) \in \Gamma_1, \\ \bar{x}_{ij} - \Theta, & \text{ha } (i, j) \in \Gamma_2. \end{cases}$$

Megmutatjuk, hogy $\mathbf{X}^*$ folyam, azaz lehetséges megoldása a (6.1) feladatnak. A definíciókból következik, hogy $0 \leq x^*_{ij} \leq c_{ij}$ teljesül minden $i \neq j \in \{1, \ldots, n\}$ indexpárra, azaz a (6.1) feladat második feltételcsoportját $\mathbf{X}^*$ kielégíti. Most legyen $i \in \{2, \ldots, n-1\}$ tetszőleges. Ha az i csúcspont nem szerepel a tekintett $\mathcal{P}$ útban, akkor $\sum_{(t,i)\in E} x^*_{ti} = \sum_{(i,t)\in E} x^*_{it}$ teljesül, mivel ezen élekre $\bar{x}_{ij} = x^*_{ij}$ és $\bar{\mathbf{X}}$ lehetséges megoldás. Ezek után tegyük fel, hogy az i csúcs szerepel a $\mathcal{P}$ irányítatlan útban. Az i csúcshoz a $\mathcal{P}$ útban kapcsolódó élek irányítását illetően az alábbi 6.1. ábrán megadott négy eset lehetséges:



6.1. ábra. Az élek lehetséges irányításai a $\mathcal{P}$ irányítatlan útban.

A 6.1. ábra alapján nyilvánvaló, hogy a tekintett i csúcsra az értékek változtatása nem rontja el azt a tulajdonságot, hogy az i csúcsba bemenő élekhez tartozó értékek összege megegyezik az i csúcsból kivezető élekhez tartozó értékek összegével. Következésképpen, $\mathbf{X}^*$ kielégíti a (6.1) feladat harmadik feltételcsoportját. Végül vizsgáljuk az első feltétel teljesülését. Mivel az $(1, i_1)$ és (i_{k-1}, n) élek mindegyike előremenő éle a $\mathcal{P}$ irányítatlan útnak, ezért

$$\sum_{(1,t)\in E} x^*_{1t} = \Theta + \sum_{(1,t)\in E} \bar{x}_{1t} = \Theta + \sum_{(t,n)\in E} \bar{x}_{tn} = \sum_{(t,n)\in E} x^*_{tn},$$

azaz $\mathbf{X}^*$ kielégíti a (6.1) feladat első feltételét is. Ezzel igazoltuk, hogy $\mathbf{X}^*$ folyam, azaz kielégíti (6.1) feltételrendszerét.

A fenti egyenlet alapján $\sum_{(1,t)\in E} x^*_{1t} = \Theta + \sum_{(1,t)\in E} \bar{x}_{1t}$, amiből $\Theta > 0$ miatt adódik, hogy az új $\mathbf{X}^*$ folyam értéke nagyobb, mint a korábbi $\bar{\mathbf{X}}$ folyamhoz tartozó érték. Ezzel a 6.1. tétel bizonyítását befejeztük.

A 6.1. tétel alapján felépíthető egy olyan eljárás, amelynél kiindulunk egy adott folyamból, ez például lehet a 0-mátrix is, majd az adott folyamhoz keresünk növelő utat és ennek alapján áttérünk egy nagyobb értékű folyamra. Ezek után az új folyamot tekintve aktuális folyamnak, ehhez keresünk növelő utat. Az eljárást addig folytatjuk, amíg az aktuális folyamhoz nem létezik növelő út.

A vázolt eljárással kapcsolatosan a következő kérdéseket lehet felvetni és megválaszolni.

(1) Véget ér-e véges lépésben az eljárás?

(2) Adott $\bar{\mathbf{X}}$ mátrixról miként lehet eldönteni, hogy tartozik-e hozzá növelő út? Ha létezik növelő út $\bar{\mathbf{X}}$-ra, akkor hogyan lehet egy növelő utat meghatározni?

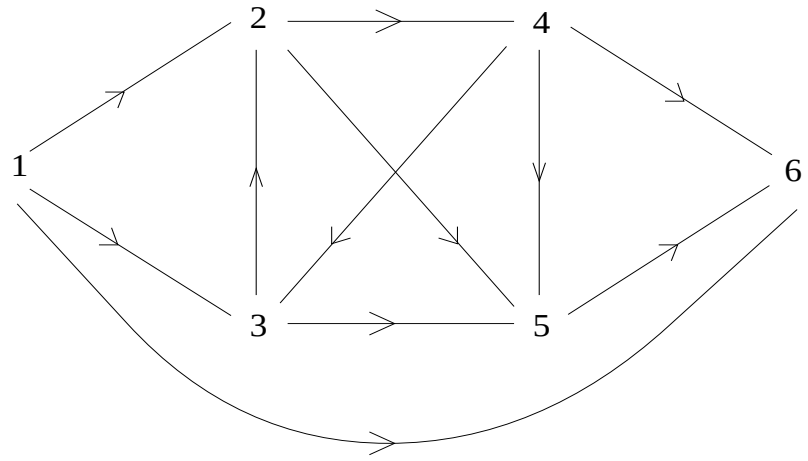
(3) Mit állíthatunk egy olyan folyamról, amelyre nem létezik növelő út?

A továbbiakban ezen kérdésekre fogunk választ adni. Elsőként a (3) kérdést vizsgáljuk. Ehhez szükségünk van bizonyos fogalmakra.

Legyen $\mathcal{G} = (\{1, \ldots, n\}, E)$ egy összefüggő irányított gráf, továbbá legyen $\emptyset \neq S \subset \{1, \ldots, n\}$ és $T = \{1, \ldots, n\} \setminus S$. Akkor az $A = (E \cap (S \times T)) \cup (E \cap (T \times S))$ élhalmazt *vágásnak* nevezzük. Szemléletesen az A élhalmaz az összes S-ből T-be vezető és az összes T-ből S-be vezető élet tartalmazza. Az elnevezést az indokolja, hogy törölve a $\mathcal{G}$ gráfból az A-beli éleket, az előálló új gráf legalább két komponenst tartalmaz, nevezetesen bármely $s \in S$ és $t \in T$ csúcspárra s és t az új gráf különböző komponenseiben lesznek.

A vágást az alábbi példával illusztráljuk.

6.1. példa. Legyen $\mathcal{G} = (\{1, \ldots, 6\}, E)$, $S = \{1, 2, 3\}$ és az élek $E = \{(1,2), (1,3), (1,6), (2,4), (2,5), (3,2), (3,5), (4,3), (4,5), (4,6), (5,6)\}$. $\mathcal{G}$ grafikus reprezentációját mutatja a 6.2. ábra.



6.2. ábra. A 6.1. példa gráfjának grafikus reprezentációja.

A tekintett példában $T = \{4, 5, 6\}$ és az $A = \{(1,6), (2,4), (2,5), (3,5), (4,3)\}$ élhalmaz a megfelelő vágás.

A vágás fogalma fontos szerepet játszik a következő állítás bizonyításában.

6.2. tétel. *Ha az $\bar{\mathbf{X}}$ folyamhoz nem létezik növelő út, akkor a hozzátartozó folyamérték maximális, azaz $\bar{\mathbf{X}}$ optimális megoldása a* (6.1) *feladatnak.*

Bizonyítás. A növelő út fogalmát értelemszerűen ki lehet terjeszteni a forrásra és tetszőleges csúcsra vonatkozóan. Jelölje S azon csúcspontok halmazát, amelyekbe vezet a forrásból, azaz az 1 csúcsból növelő út $\mathcal{G}$-ben, továbbá vegyük az 1 csúcsot is S-hez. A tétel feltétele szerint $n \notin S$, mivel nem létezik 1-ből n-be vezető növelő út. Legyen $T = \{1, \ldots, n\} \setminus S$ és A az ezen S, T páros által meghatározott vágás, azaz az összes S-ből T-be vezető és az összes T-ből S-be vezető élek halmaza. Mivel $\mathcal{G}$ összefüggő, ezért A nemüres. Akkor az egész hálózaton átbocsájtható anyagmennyiség nem haladhatja meg az S halmazból a T halmazba vivő élek átbocsájtóképességeinek összegét, mivel az anyagmennyiségnek át kell mennie az $E \cap (S \times T)$ élhalmazon. Tehát a hálózat átbocsájtóképessége nem nagyobb, mint a $d = \sum_{(i,j)\in A, i\in S, j\in T} c_{ij}$ érték, azaz $z(\mathbf{X}^*) \leq d$ bármely $\mathbf{X}^*$ folyamra.

Vizsgáljuk most az $\bar{\mathbf{X}}$ folyam értékét. Erre $z(\bar{\mathbf{X}}) = \sum_{(1,t)\in E} \bar{x}_{ij} = \sum_{(t,n)\in E} \bar{x}_{tn}$. Ennek az anyagmennyiségnek át kell haladnia S és T között. Tekintve az S és T közötti éleket, az alábbi egyenlőségnek kell fennállnia:

$$z(\bar{\mathbf{X}}) = \sum_{\substack{(i,j)\in A,\\ i\in S,\, j\in T}} \bar{x}_{ij} - \sum_{\substack{(u,v)\in A,\\ u\in T,\, v\in S}} \bar{x}_{uv}.$$

Vizsgáljuk a fenti összegek tagjait. Elsőként tekintsünk egy olyan $(i,j) \in A$ élet, amelyre $i \in S$ és $j \in T$. Az S definíciója szerint vezet az 1 csúcsból i-be növelő út. Ekkor viszont $\bar{x}_{ij} = c_{ij}$-nek kell teljesülnie, mivel ellenkező esetben vezetne 1-ből j-be is növelő út, ami ellentmondás. Tehát minden $(i,j) \in A$, $i \in S$, $j \in T$ élre $\bar{x}_{ij} = c_{ij}$, azaz a fenti kifejezés első tagjában szereplő $\bar{x}_{ij}$-k rendre megegyeznek a c_{ij} értékekkel. Tekintsünk most egy olyan $(u,v) \in A$ élet, amelyre $u \in T$ és $v \in S$. Akkor $v \in S$ miatt ismét létezik 1-ből v-be vezető növelő út. De akkor az $\bar{x}_{uv} = 0$ egyenlőségnek kell teljesülnie, mivel ellenkező esetben lenne 1-ből u-ba vezető növelő út, ami ellentmondás. Következésképpen, a fenti kifejezés második tagjában szereplő $\bar{x}_{ij}$-k rendre megegyeznek 0-val. Így

$$z(\bar{\mathbf{X}}) = \sum_{\substack{(i,j)\in A,\\ i\in S,\, j\in T}} \bar{x}_{ij} - \sum_{\substack{(u,v)\in A,\\ u\in T,\, v\in S}} \bar{x}_{uv} = \sum_{\substack{(i,j)\in A,\\ i\in S,\, j\in T}} c_{ij} = d,$$

azaz $z(\bar{\mathbf{X}}) = d$, amivel a 6.2. tételt igazoltuk.

A fenti tétellel teljes mértékben megválaszoltuk a (3) kérdést, azaz ha egy adott $\bar{\mathbf{X}}$ folyamhoz nem létezik növelő út, akkor az illető folyam optimális megoldás, amit szokásos *maximális folyamnak* nevezni.

Vizsgáljuk ezek után a (2) kérdést. Nyilvánvaló, hogy az irányítatlan utak véges száma miatt a növelő út egzisztenciája eldönthető, és amennyiben létezik az adott folyamra növelő út, akkor egy ilyen út meg is határozható. Gyakorlati szempontból azonban a növelő utak ilyen vizsgálata nagyon nehézkes és nagymennyiségű számítást igényel. A következőkben egy jóval hatékonyabb eljárással fogunk megismerkedni. Az eljárást L. R. Ford és D. R. Fulkerson [59] dolgozták ki 1957-ben.

Ford és Fulkerson eljárása ([59])

Előkészítő rész

Lássuk el az 1 csúcsot a $(-, +\infty)$ címkével és adjuk r-nek a 0 értéket.

Iterációs rész (r-edik iteráció)

(Bizonyos csúcsok el vannak látva címkékkel, mások nem, továbbá a címkézett csúcsokból bizonyosak már átvizsgálásra kerültek.)

- *1. lépés.* Ha minden címkézett csúcsot átvizsgáltunk, akkor vége az eljárásnak; a tekintett $\bar{\mathbf{X}}$ folyamhoz nem létezik növelő út. Ellenkező esetben a 2. lépés következik.

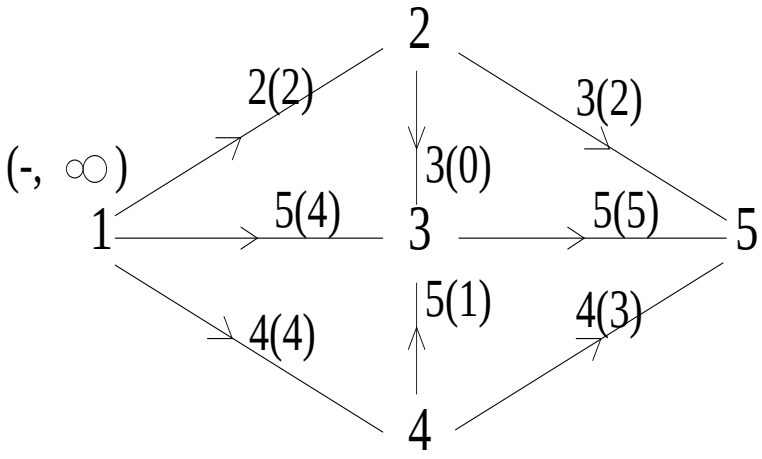
- *2. lépés.* Válasszunk egy címkézett, de nem átvizsgált i csúcsot és vizsgáljuk át a következők szerint. Minden $(i,t) \in E$ élre, ha $\bar{x}_{it} < c_{it}$ és a t csúcs címkézetlen, akkor adjuk a t csúcsnak az (i^+, δ_t) címkét, ahol $\delta_t = \min\{\delta_i, c_{it} - \bar{x}_{it}\}$. Továbbá minden $(t,i) \in E$ élre, ha $\bar{x}_{ti} > 0$ és a t csúcs címkézetlen, akkor lássuk el a t csúcsot az (i^-, δ_t) címkével, ahol $\delta_t = \min\{\delta_i, \bar{x}_{ti}\}$. Amennyiben a címkézések során a nyelő, azaz az n csúcs címkét kap, akkor vége az eljárásnak; a tekintett $\bar{\mathbf{X}}$ folyamhoz létezik növelő út. Az átvizsgálás befejeztével tekintsük az i csúcsot átvizsgáltnak, növeljük r értékét 1-gyel, és térjünk rá a következő iterációra.

A címkék alapján a növelő utat a következők szerint határozhatjuk meg. Tegyük fel, hogy a nyelő, az n csúcspont meg lett címkézve, és legyen a címkéje (i_k^+, δ). Akkor n őse a növelő útban i_k lesz. Most véve i_k címkéjét,

ebben is szerepel egy szögpont, ez lesz az i_k őse a keresett útban. Így folytatva az eljárást, a nyelőből visszafele haladva, megkapjuk a növelő utat.

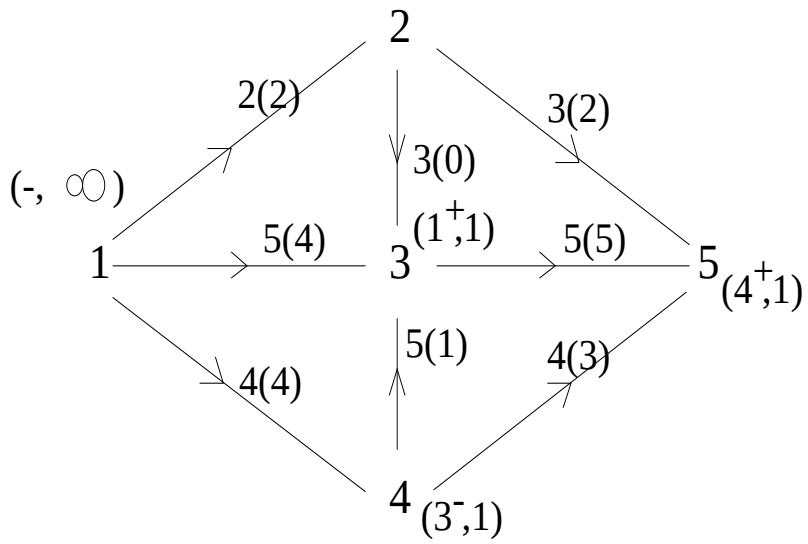
Az eljárás helyességének igazolása előtt a módszert a következő példán illusztráljuk.

6.2. példa. Legyen $\mathcal{G} = (\{1, \ldots, 5\}, E)$, ahol az E-ben levő élek: $(1,2)$, $(1,3)$, $(1,4)$, $(2,3)$, $(2,5)$, $(3,5)$, $(4,3)$, $(4,5)$, továbbá $c_{12} = 2$, $c_{13} = 5$, $c_{14} = 4$, $c_{23} = 3$, $c_{25} = 3$, $c_{35} = 5$, $c_{43} = 5$, $c_{45} = 4$. Tekintsük azt az $\bar{\mathbf{X}}$ folyamot, amelyre $\bar{x}_{12} = 2$, $\bar{x}_{13} = 4$, $\bar{x}_{14} = 4$, $\bar{x}_{23} = 0$, $\bar{x}_{25} = 2$, $\bar{x}_{35} = 5$, $\bar{x}_{43} = 1$, $\bar{x}_{45} = 3$, és a fel nem sorolt indexekre legyen $\bar{x}_{ij} = 0$. A tekintett problémát mutatja a 6.3. ábra, ahol az élek átbocsájtóképessége mellett rendre zárójelben feltüntettük a megfelelő $\bar{x}_{ij}$ értékeket.



6.3. ábra. A 6.2. példa hálózata az $\bar{\mathbf{X}}$ folyammal.

Első lépésként a forrást ellátjuk a $(-, \infty)$ címkével. Ezt a 6.3. ábrán meg is tettük. Ekkor egyetlen csúcspontnak, nevezetesen a forrásnak van átvizsgálatlan címkéje, így az iterációs részben ezt a csúcsot kell átvizsgálni. Egyetlen élre, az $(1,3)$-ra teljesül az előírt feltétel, így a 3 csúcs az $(1^+, 1)$ címkét kapja. Ekkor a forrás már átvizsgált csúcs lesz, és ismét egyetlen olyan csúcsunk lesz, a 3 csúcs, amelynek van címkéje és nincs átvizsgálva. A következő iterációs lépésben ezt a csúcsot vizsgáljuk át, és ekkor a 4 csúcs kapja a $(3^-, 1)$ címkét. Ezt követően pedig a nyelő kapja a $(4^+, 1)$ címkét. Ezzel vége az eljárásnak, a növelő útban 5 őse 4, 4-et 3 előzi meg, és 3 őse 1. A befejezett eljárás utáni helyzetet mutatja a 6.4. ábra.



6.4. ábra. A 6.2. példa hálózata a címkézési eljárás végén.

A tekintett példával kapcsolatban megjegyezzük még, hogy amennyiben a 6.1. tétel bizonyításában megadottak szerint megváltoztatjuk az $\bar{\mathbf{X}}$ mátrix elemeit és az új folyamot $\mathbf{X}^*$-gal jelöljük, akkor végrehajtva $\mathbf{X}^*$-ra a címkézési eljárást, azt kapjuk, hogy $\mathbf{X}^*$-hoz nem létezik növelő út, azaz $\mathbf{X}^*$ egy optimális megoldás. Konkrétan, a forrás átvizsgálása során nem tudunk egyetlen csúcspontot sem megcímkézni, így az adódik, hogy minden címkézett csúcsot átvizsgáltunk.

Az eljárás helyességének igazolása

Jelölje $\mathcal{G} = (\{1, \ldots, n\}, E)$ a hálózat gráfját. Tételezzük fel, hogy az eljárás során az i csúcsponthoz egy bővebb (j^+, δ_i, k_i) címkét rendelünk, ahol a k_i-t a következőképpen definiáljuk:

(a) a forrás címkéje $(-, \infty, 0)$,

(b) a (j^+, δ_i, k_i) vagy (j^-, δ_i, k_i) címkéjű i csúcs átvizsgálása során minden címkézetlen t csúcsra $k_t = k_i + 1$, ha a t csúcs címkét kap ebben az átvizsgálásban.

Most a címkék k-val jelölt harmadik komponense szerinti teljes indukcióval bizonyítjuk, hogy bármely címkével rendelkező csúcspontba vezet a forrásból olyan növelő út, amelyben az élek száma megegyezik a címke harmadik

komponensével, továbbá a tekintett folyam értéke a címke második komponensében megadott értékkel növelhető ezen növelő út mentén.

Ha $k = 1$, akkor az állítás nyilvánvaló. Most tegyük fel, hogy $k > 1$, és $k-1$-re teljesül az állítás. Legyen a tekintett csúcs j és jelölje a címkéjét (i^+, δ_j, k) vagy (i^-, δ_j, k). Akkor a j csúcs az i csúcs átvizsgálása során kapta a címkéjét, így az i csúcs címkéje $(r^*, \delta_i, k-1)$, ahol $*$ lehet a $+$ és $-$ előjelek bármelyike, továbbá vagy $(i,j) \in E$ és $\bar{x}_{ij} < c_{ij}$, vagy $(j,i) \in E$ és $\bar{x}_{ji} > 0$ teljesül. Az indukciós feltevés alapján a forrásból vezet az i csúcsba növelő út és ezen út mentén a tekintett folyam értéke δ_i-vel növelhető. De akkor folytatva ezt az utat az (i,j) vagy (j,i) éllel, nyilvánvalóan egy, a forrásból a j csúcsba vezető növelő utat kapunk, továbbá a tekintett folyam értéke δ_j-vel növelhető ezen út mentén, azaz teljesül k-ra is az állítás.

Az ismertetett eljárás megoldást ad a (2) pont alatt felvetett mindkét kérdésre, így csak az (1) alatt felvetett probléma megoldása maradt nyitva.

Tulajdonképpen (1) megválaszolását az élekhez rendelt valós számok milyensége befolyásolja. Ha ezen számok mindegyike egész, és kiindulunk egy egész értékeket tartalmazó $\bar{\mathbf{X}}$ folyamból, akkor az ismertetett eljárásban a δ értékek is egészek lesznek. Ekkor az eljárás során előálló $\bar{\mathbf{X}}$, $\mathbf{X}^{(1)}$, $\mathbf{X}^{(2)}, \ldots$ folyamsorozat minden tagja egész lesz. Másrészt, mivel minden δ pozitív és egész, a folyamsorozathoz tartozó folyamértékek egészek egy szigorúan monoton növekedő sorozatát alkotják, amely felülről korlátos. Egy alkalmas korlát például a $\sum_{(1,t)\in E} c_{1t}$ érték. Következésképp, az eljárás legfeljebb $\sum_{(1,t)\in E} c_{1t}$ lépésben véget ér, és az optimális megoldás egész, amennyiben az induló $\bar{\mathbf{X}}$ egész. Ez utóbbi viszont biztosítható, például választhatjuk $\bar{\mathbf{X}}$-t a 0-mátrixnak. A fenti észrevételekből adódik a folyamproblémák alábbi fontos tulajdonsága.

6.3. tétel. *Ha a* (6.1) *problémában a* c_{ij} $(1 \le i \ne j \le n)$ *értékek mindegyike egész, akkor létezik a problémának egész számokból álló optimális megoldása.*

Bizonyítás. A korábbiakban megismert eljárás a 6.3. tétel egy konstruktív bizonyítása.

A fentiekkel kapcsolatban vegyük észre, hogy az elmondottak akkor is érvényesek, ha a c_{ij} együtthatók valamely alkalmas w egységelemnek a többszörösei. Ugyanis ebben az esetben minden lépésben legalább w-vel nő a folyam értéke. Ennek egy speciális esete, amikor minden c_{ij} racionális. Ekkor egy alkalmas egység lehet a c_{ij}-kben szereplő nevezők legkisebb közös többszörösének a reciproka.

Sajnos az eljárás végessége nem igaz általános esetben. Ford és Fulkerson a [60] munkában megadott egy olyan példát, amelyre az eljárás nem ér véget véges lépésben, hanem minden határon túl folytatódik. Szerencsére nem ilyen rossz a helyzet, ugyanis a megismert eljárás kis módosításával elérhető annak végessége. Egy ilyen módosítást adott meg 1972-ben J. Edmonds és R. M. Karp [49]. Ezzel fogunk megismerkedni a továbbiakban. A módosítás szemléletes jelentése a következő. Az iterációs részben a címkézett, de át nem vizsgált csúcsok közül nem tetszőlegesen választunk, hanem olyan sorrendben, ahogy a csúcsok el lettek látva címkével. Ennek megvalósítására nagyon jól használható a címkézési eljárás helyességének igazolásakor bevezetett bővített címke harmadik komponense. Ezt használva, egy alkalmas módosítás lesz, ha a címkézett, de át nem vizsgált csúcsok közül olyant választunk, melynek bővített címkéjében a harmadik komponens minimális. Ilyen módosítással már bármilyen valós c_{ij} értékek mellett véges lépésben véget ér az eljárás.

A következőkben a folyamproblémák egy fontos tételét igazoljuk, amely a *maximális folyam, minimális vágás tétele* néven ismeretes. Ehhez szükségünk lesz az alábbi definícióra.

Az S és $T = \{1, \ldots, n\} \setminus S$ halmazok által meghatározott A vágáshoz kapacitást tudunk rendelni a következők szerint. Az A *vágás kapacitásán* az S-ből a T-be vezető élek átbocsájtóképességeinek az összegét értjük. Formálisan ez a kapacitás a $\sum_{(i,j)\in E\cap S\times T} c_{ij}$ összeggel egyenlő. Használni fogunk még egy további fogalmat. Legyen $s \neq t \in \{1, \ldots, n\}$ két tetszőleges csúcs. A $\mathcal{G}$ gráf olyan vágásait, amelyeket meghatározó S halmazokra $s \in S$ és $t \notin S$ teljesül, (s, t) *vágásoknak* nevezzük.

6.4. tétel ([58] Maximális folyam, minimális vágás tétele). *A maximális folyamérték megegyezik az* $(1, n)$ *vágások minimális kapacitásával.*

Bizonyítás. Legyen $\bar{\mathbf{X}}$ egy maximális értékű folyam. Tekintsünk egy tetszőleges $(1, n)$ vágást. Jelölje S a vágást meghatározó halmazt. Akkor $1 \in S$ és $n \notin S$. Legyen $T = \{1, \ldots, n\} \setminus S$. Ekkor S-ből T-be csak az $S \times T$ halmazban levő éleken történhet áramlás, így a hálózaton áthaladó anyagmennyiség nem lehet nagyobb, mint $\sum_{(i,j)\in E\cap(S\times T)} c_{ij}$, ami viszont pontosan a tekintett vágás kapacitása. Tehát $z(\bar{\mathbf{X}}) \leq \sum_{(i,j)\in E\cap(S\times T)} c_{ij}$. Mivel tetszőleges $(1, n)$ vágást tekintettünk, ezért már csak azt kell igazolni, hogy létezik olyan $(1, n)$ vágás, amelynek a kapacitása pontosan $z(\bar{\mathbf{X}})$-sal egyenlő. Ez konstruktívan igazolható. Hajtsuk végre a Ford-Fulkerson féle címkézési eljárást, és jelölje S azon csúcsok halmazát, amelyekbe vezet az 1 forrásból növelő út $\bar{\mathbf{X}}$-ra nézve. Mivel $\bar{\mathbf{X}}$ maximális értékű folyam, ezért nem

létezik hozzá a forrásból a nyelőbe vezető növelő út, de akkor $n \notin S$. Legyen $T = \{1, \ldots, n\} \setminus S$, és legyen A az S halmaz által meghatározott vágás. Ekkor A egy $(1, n)$ vágás. Másrészt a 6.2. tétel bizonyításából tudjuk, hogy $z(\bar{\mathbf{X}}) = \sum_{(i,j) \in A \cap (S \times T)} c_{ij}$, azaz $z(\bar{\mathbf{X}})$ megegyezik a tekintett $(1, n)$ vágás kapacitásával. Ezzel a 6.4. tételt igazoltuk.

A 6.4. tétellel kapcsolatban megemlítjük, hogy az interpretálható a dualitási tétel egy szép alkalmazásaként. Ilyen megközelítésben a (6.1) feladat tekinthető a primál feladatnak, melynek lehetséges megoldásai a folyamok. A (6.1) feladat duálisában pedig az $(1, n)$ vágások alkotják a lehetséges megoldásokat. Akkor a 6.4. tétel pontosan az erős dualitási tételnek egy speciális alkalmazása.

A tekintett folyamproblémának több különböző általánosítása van. A továbbiakban ezeket vázoljuk érintőlegesen.

Folyamprobléma csúcs- és élkapacitásokkal

Ebben a legegyszerűbben kezelhető általánosításban a hálózatban nemcsak az élekhez, hanem a csúcsokhoz is számok vannak rendelve, amelyek az illető csúcs átbocsájtóképességeként interpretálhatók. Ekkor a lehetséges folyamoknak az élkapacitásokra vonatkozó feltételek mellett a csúcskapacitásokra vonatkozó feltételeket is ki kell elégíteni, és a kibővített feltételrendszer mellett keresünk egy maximális értékű folyamot.

Az új probléma egyszerűen visszavezethető a (6.1) feladatra a következő módon. Minden c_i kapacitású i csúcsot helyettesítsünk a hálózatban egy (i, i') éllel, amelynek legyen a kapacitása c_i. Továbbá az (i, t) alakú c_{it} kapacitású éleket töröljük a hálózatból és vegyünk fel helyettük az (i', t) éleket c_{it} kapacitással. Egyszerűen belátható, hogy az így előálló folyamprobléma már (6.1) alakú, és ekvivalens a másik feladattal abban az értelemben, hogy a két feladat optimális megoldásai közvetlenül származtathatók egymásból.

Minimális költségű folyamok

Ebben a modellben azt tételezzük fel, hogy minden élhez a kapacitás mellett egy további mennyiség is adott, amely az illető élen áthaladó egységnyi anyagmennyiség áthaladási költségeként interpretálható. Az egyszerűbb szóhasználat érdekében ezt a költséget gyakran az illető *él költségének* nevezzük. Ha ezeket a költségegyütthatókat a_{ij}-vel jelöljük, ahol $a_{ij} = 0$, ha (i, j)

nem éle a hálózatnak, akkor egy $\bar{\mathbf{X}}$ folyam költsége $\sum_{i,j\in\{1,\dots,n\}} a_{ij}\bar{x}_{ij}$. Ezek után a feladat a következő:

határozzunk meg az előírt ν értékű folyamok közül egy minimális költségűt.

A minimális költségű folyamok témaköre kiterjedt vizsgálatokhoz vezetett, aminek oka részben abban rejlik, hogy a hozzárendelési és szállítási feladatok visszavezethetők ilyen problémákra.

A következőkben vázoljuk a probléma egy megoldási módszerét. Ehhez szükségesek bizonyos fogalmak. Elsőként kiterjesztjük a növelő út fogalmát. Egy, az i csúcsból a j csúcsba vezető irányítatlan utat az $\bar{\mathbf{X}}$ folyamra nézve növelő útnak nevezünk, ha minden előremenő (k,l) élére $\bar{x}_{kl} < c_{kl}$ és minden (u,v) hátramenő élére $\bar{x}_{uv} > 0$ teljesül. Egy *növelő út költségét* a következőképpen definiáljuk. Az út előremenő élei költségeinek összegéből kivonjuk az út hátramenő élei költségeinek összegét. Ennek a szemléletes jelentése a következő. Ha a növelő út mentén egységnyi mennyiséggel növeljük a folyamhoz tartozó $\bar{x}_{ij}$ értékeket, akkor az út költsége pontosan a folyamváltozás költségét adja meg. Azt mondjuk, hogy az $\bar{\mathbf{X}}$ folyamra vonatkozóan létezik *növelő kör*, ha vannak olyan $i \neq j \in \{1,\dots,n\}$ csúcsok, hogy i-ből vezet j-be és j-ből vezet i-be növelő út. A *növelő kör költségén* a két növelő út költségeinek összegét értjük. Vegyük észre, hogy a növelő kör létezése nem eredményezi azt, hogy az $\bar{\mathbf{X}}$ folyam értéke növelhető, csupán azt jelenti, hogy az $\bar{\mathbf{X}}$ folyamban cirkuláló kör mentén az $\bar{x}_{ij}$ értékek megváltoztatásával képezhetünk egy másik, de az $\bar{\mathbf{X}}$-sal azonos értékű folyamot. Ez arra lesz jó, hogy rögzített folyamérték mellett csökkentsük a folyam költségét. Ezek után egy megoldási módszer a következő:

Eljárás ν értékű minimális költségű folyam meghatározására

Előkészítő rész

- *1.1. lépés.* Határozzunk meg egy $\mathbf{X}^{(0)}$ folyamot, melynek ν^* értéke nem nagyobb, mint ν. Legyen $m = 0$ és térjünk rá az 1.2. lépésre.

- *1.2. lépés.* Keressünk negatív költségű növelő kört $\mathbf{X}^{(m)}$-re. Ha ilyen nincs, akkor folytassuk az eljárást az 1.4. lépéssel. Ellenkező esetben az 1.3. lépés következik.

- *1.3. lépés.* Képezzük a növelő kör előremenő (k,l) éleire a $c_{kl} - x_{kl}^{(m)}$ értékek Θ_1-gyel jelölt minimumát és a hátramenő (u,v) éleire az $x_{uv}^{(m)}$ értékek Θ_2-vel jelölt minimumát, majd legyen $\Theta = \min\{\Theta_1, \Theta_2\}$. A növelő kör előremenő (k,l) éleire legyen $x_{kl}^{(m+1)} = x_{kl}^{(m)} + \Theta$ és a növelő

kör hátramenő (u, v) éleire legyen $x_{uv}^{(m+1)} = x_{uv}^{(m)} - \Theta$. A kimaradó (i, j) indexekre legyen $x_{ij}^{(m+1)} = x_{ij}^{(m)}$. Növeljük m értékét 1-gyel, és folytassuk az eljárást az 1.2. lépéssel.

- *1.4. lépés* Legyen $\nu_1 = \nu^*$, $\mathbf{X}^{(1)} = \mathbf{X}^{(m)}$ és $r = 1$, majd térjünk rá az iterációs eljárásrészre.

  Iterációs rész (r-edik iteráció)

- *2.1. lépés.* Ha $\nu_r = \nu$, akkor vége az eljárásnak; $\mathbf{X}^{(r)}$ minimális költségű ν értékű folyam. Ellenkező esetben a 2.2. lépés következik.

- *2.2. lépés.* Határozzunk meg egy minimális költségű, a forrásból a nyelőbe vezető növelő utat. Ha ilyen nincs, akkor vége az eljárásnak; a problémának nincsen lehetséges megoldása. Különben folytassuk az eljárást a 2.3. lépéssel.

- *2.3. lépés.* Jelölje Θ a növelő út mentén felvehető plusz anyagmennyiséget. Legyen $\Theta' = \min\{\Theta, \nu - \nu_r\}$. Változtassuk meg az $\mathbf{X}^{(r)}$ folyam értékét Θ'-vel a növelő út mentén, és az új folyamot jelölje $\mathbf{X}^{(r+1)}$. Legyen $\nu_{r+1} = \nu_r + \Theta'$. Növeljük r értékét 1-gyel, majd térjünk rá a következő iterációs lépésre.

Az eljárás helyessége az alábbi tételeken alapul, amelyek igazolásától eltekintünk.

6.5. tétel. *Egy ν értékű $\bar{\mathbf{X}}$ folyam akkor és csak akkor minimális költségű, ha nem létezik hozzá negatív költségű növelő kör.*

6.6. tétel ([29], [99]). *Legyen $\bar{\mathbf{X}}$ egy ν értékű minimális költségű folyam, továbbá P egy, a forrásból a nyelőbe vezető minimális költségű növelő út, amely mentén Θ-val növelhető az $\bar{\mathbf{X}}$ folyam értéke. Akkor a P út mentén, bármely $0 < \delta < \Theta$ értékkel növelve az $\bar{\mathbf{X}}$ folyamot, az előálló új $\nu + \delta$ értékű folyam szintén minimális költségű lesz.*

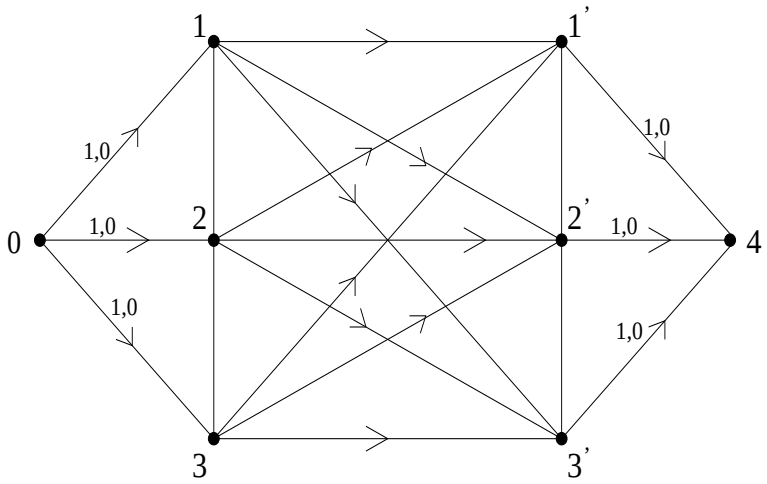
A teljességhez még hozzátartozna az, hogy megadjunk olyan eljárásokat, amelyek lehetővé teszik adott értékű adott folyamhoz

(1) negatív költségű növelő kör létezésének eldöntését, és pozitív esetben egy ilyen növelő kör meghatározását, valamint

(2) a forrásból a nyelőbe vezető növelő utak közül egy minimális költségű növelő út meghatározását.

Itt nem térünk ki ezen eljárások ismertetésére, ilyen eljárások találhatók a [54] és [122] munkákban. Ezzel szemben két példán illusztráljuk, hogy miként vezethető vissza a hozzárendelési feladat és a zárt szállítási feladat a tekintett folyamproblémára.

6.3. példa. Tekintsünk egy 3×3-as hozzárendelési feladatot, amelynek költségmátrixa legyen (c_{ij}). Konstruáljuk meg a tekintett hozzárendelési feladathoz a 6.5. ábrán megadott hálózatot, ahol az éleken elsőként az átbocsájtóképességet, majd másodikként az illető él költségét tüntettük fel. A forrás most 0, a nyelő pedig 4, továbbá minden címkézetlen (i, j') élre az átbocsájtóképesség 3 és a költség c_{ij}.

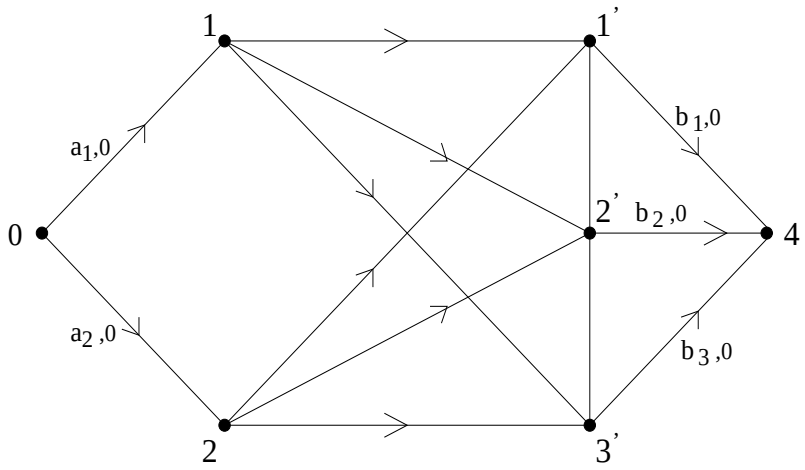


6.5. ábra. A 6.3. példa hozzárendelési feladatát leíró folyamprobléma.

Nyilvánvaló, hogy a 3 értékű minimális költségű egészértékű folyam szolgáltatja a hozzárendelési feladat egy optimális megoldását.

6.4. példa. Tekintsünk egy olyan szállítási feladatot, amelyben két feladó hely van, rendre a_1 és a_2 kapacitásokkal, továbbá három felvevő hely, rendre a b_1, b_2, b_3 kapacitásokkal. Továbbá az egységnyi anyagmennyiségre vonatkozó szállítási költségek a (c_{ij}) mátrixszal vannak megadva. Konstruáljuk meg a tekintett szállítási feladathoz a 6.6. ábrán megadott hálózatot, ahol az éleken ismét az átbocsájtóképességet, majd az illető él költségét tüntettük fel. Itt a forrás 0 és a nyelő 4, továbbá minden címkézetlen (i, j') élre az átbocsájtóképesség $a_1 + a_2$ és a költség c_{ij}.

Egyszerűen meggondolható, hogy az $a_1 + a_2$ értékű minimális költségű folyam szolgáltatja a tekintett szállítási feladat egy optimális megoldását.



6.6. ábra. A 6.4. példa szállítási feladatát leíró folyamprobléma.

Veszteséges és nyereséges hálózatok

Az ilyen modellekben a hálózat éleire nem teljesül az a feltétel, miszerint az élen az anyagmennyiség mennyiségi változás nélkül áthalad, hanem az áthaladás során előfordulhat növekedés, azaz nyereség, vagy csökkenés, azaz veszteség. Formálisan, minden (i, j) élhez tartozik egy nemnegatív m_{ij} mennyiség, amely megadja, hogy amennyiben az (i, j) élbe x_{ij} mennyiségű anyagot viszünk be az i pontnál, akkor $m_{ij}x_{ij}$ anyagmennyiséget nyerünk az él j végpontjánál. Ebben az esetben a rendszerbe a forráson bemenő anyagmennyiség $\nu_1 = \sum_{(1,t)\in E} x_{1t}$ és a rendszerből a nyelőn távozó anyagmennyiség $\nu_n = \sum_{(t,n)\in E} m_{tn}x_{tn}$. A kettő különbsége a folyam vesztesége. Az ilyen modellekben főképp a minimális veszteségű folyamok meghatározása a cél, azaz rögzített ν_1 mellett olyan folyamot keresünk amelyre ν_n maximális, vagy rögzített ν_n mellett olyan folyamot keresünk, melyre ν_1 minimális.

Többkimenetű és többtermékes folyamok

Az eddigiekben feltételeztük, hogy egyféle anyag áramlik stacionáriusan a hálózatban, amelyben kitüntettünk egy forrást és egy nyelőt, és ezen párosra vizsgáltuk az anyag áramlását. Abban az esetben, ha megengedünk több forrást és több nyelőt, az úgynevezett *többkimenetű folyamproblémához* jutunk. A másik kézenfekvő lehetőség megengedni azt, hogy szimultán több-

féle anyag is áramolhasson a hálózatban. Ilyenkor *többtermékes folyamprobléméról* beszélünk.

Amint azt az eddigiekben felsorolt modellek mutatják, a hálózati folyamok témaköre igen gazdag. A témakör kutatói számos szép és mély eredményt értek el ezen a területen. A téma iránt érdeklődőknek ismételten ajánljuk a [38], [54], és [122] könyveket.

7. Korlátozás és szétválasztás módszere

A címben szereplő módszer egy általános kereteljárásként interpretálható, melynek különböző konkrét alkalmazásai igen hatékony eszközt szolgáltatnak a kombinatorikus optimalizálási feladatok megoldásában.

A módszer alapgondolata A. H. Land és A. G. Doig [119] munkájában lelhető fel első ízben. A szerzők a vegyes egészértékű lineáris programozási feladat megoldására adtak meg egy olyan eljárást, amely alapvetően erre a technikára épült. Maga az igen találó angol *"Branch-and-Bound"* elnevezés J. D. C. Little és társai [128] dolgozatában szerepelt először, amelyben ezen technika felhasználásával egy, az utazó ügynök probléma megoldására szolgáló eljárás került publikálásra. A korábbi speciális eseteket szintetizálva, P. Bertier és B. Roy [22] dolgozta ki a módszer első általános változatát. Azóta a korlátozás és szétválasztás módszere számos formában, az általánosság különböző szintjein nyert publikálást és alkalmazásával igen sok speciális problémához hatékony megoldási eljárások készültek.

A továbbiakban mi is több változaton keresztül fogjuk felépíteni az eljárást, az egyes változatokat speciális problémamegoldásokkal fogjuk demonstrálni.

Alapeljárás

A módszer tárgyalásához tekintsük az alábbi optimalizálási problémát:

$$\min\{z(\mathbf{x}) : \mathbf{x} \in L\} \tag{7.1}$$

ahol L azonos dimenziójú, egész koordinátájú, nemnegatív vektorok véges és nemüres halmaza.

Az adott feltételek mellett a (7.1) alatti feladatnak nyilvánvalóan létezik optimális megoldása, ugyanis véges sok célfüggvényérték minimumát keressük. Legegyszerűbb megoldási módszernek első közelítésben az tűnhetne, hogy L elemeit végigvizsgáljuk, és vesszük egy olyan elemét, amelyen $z(\mathbf{x})$ a legkisebb értéket veszi fel. Ezzel egy optimális megoldáshoz jutunk. Az ilyen típusú eljárást szokásos *leszámlálási eljárásnak* nevezni. A különböző gyakorlati problémáknál L elemszáma igen nagy lehet. Például $n \times n$-es

költségmátrixú hozzárendelési feladat esetén $|L| = n!$. Így célszerű lehetőség szerint a teljes leszámlálást elkerülni, és minél kevesebb lehetséges megoldást megvizsgálni. Ez utóbbi célokat igyekszik realizálni a korlátozás és szétválasztás módszere.

Az eljáráshoz szükséges két függvény. Ezek egyike az úgynevezett *szétválasztási függvény,* amely az L halmaz tetszőleges $|L'| > 1$ részhalmazához hozzárendeli L' egy valódi osztályozását. A másik függvény, az úgynevezett *korlátozó függvény*, amely L tetszőleges $L' \neq \emptyset$ részhalmazához hozzárendeli a $z(\bar{\mathbf{x}})$ $(\bar{\mathbf{x}} \in L')$ függvényértékek egy alsó korlátját. Speciálisan, $L' = \{\tilde{\mathbf{x}}\}$ esetén a $z(\tilde{\mathbf{x}})$ függvényértéket. Most tegyük fel, hogy rendelkezésünkre állnak ilyen függvények, és jelölje a szétválasztási függvényt φ, a korlátozó függvényt pedig g.

Az eljárás során egy úgynevezett *leszámlálási fát* vagy *Branch-and-Bound fát* (a továbbiakban *B&B fa*) építünk fel a következők szerint.

Eljárás

Előkészítő rész. A fa gyökere legyen L. Határozzuk meg $g(L)$-t és rendeljük címkeként az L szögponthoz. Legyen $r = 1$.

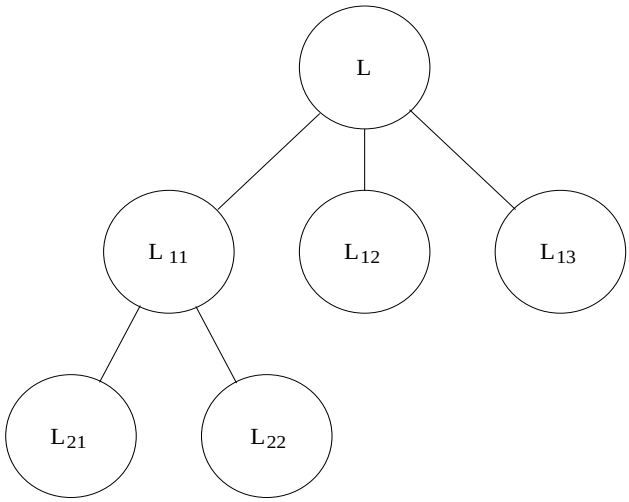
Iterációs rész (r-edik iteráció)

- *1. lépés.* Az aktuális fa levelein határozzuk meg a címkék minimumát és válasszunk ki egy minimális címkéjű L' levelet.
- *2. lépés.* Ha $L' = \{\bar{\mathbf{x}}\}$, akkor vége az eljárásnak; $\bar{\mathbf{x}}$ optimális megoldás. Ellenkező esetben a 3. lépés következik.
- *3. lépés.* Bővítsük az aktuális fát $\varphi(L')$ elemeivel, mint L' leszármazottaival, majd az új szögpontokhoz rendre számítsuk ki a korlátokat és rendeljük az illető szögpontokhoz címkeként. Ezek után növeljük r értékét 1-gyel, majd térjünk rá a következő iterációs lépésre.

Az eljárás két iterációs lépésében felépülő $B\&B$ fát szemlélteti a 7.1. ábra, ahol

$$\varphi(L) = \{L_{11}, L_{12}, L_{13}\}, \quad g(L_{11}) \leq g(L_{1j}), \quad (j = 2, 3),$$

$$\varphi(L_{11}) = \{L_{21}, L_{22}\}.$$



7.1. ábra. B&B fa két iterációs lépés után.

Az eljárás helyessége

Végesség. Vegyük észre, hogy φ definíciója alapján minden szögpontnak legfeljebb $|L|$ leszármazottja van, továbbá minden iterációs lépésben az aktuális fa leveleihez tartozó halmazok L-nek egy osztályozását alkotják. Szintén φ definíciója alapján az egymást követő osztályozások egymás valódi finomításai, így a felépíthető fa mélysége legfeljebb $|L|$. De akkor a felépíthető fa véges, amivel adódik az eljárás végessége.

Helyesség. Tegyük fel, hogy valamely iterációs lépésben L'-höz tartozik a minimális korlát és $L' = \{\bar{\mathbf{x}}\}$. Jelölje R a leveleknek megfelelő halmazok halmazát. Akkor $z(\bar{\mathbf{x}}) = g(L') \leq g(S)$ bármely $S \in R$-re. De $g(S) \leq \min\{z(\mathbf{x}) : \mathbf{x} \in S\}$, azaz $g(S) \leq z(\mathbf{x})$ minden $\mathbf{x} \in S$-re. Így $z(\bar{\mathbf{x}}) \leq z(\mathbf{x})$ bármely $\mathbf{x} \in S$-re és bármely $S \in R$-re. De akkor mivel R osztályozása L-nek, ezért $z(\bar{\mathbf{x}}) \leq z(\mathbf{x})$ bármely $\mathbf{x} \in L$-re, amivel igazoltuk, hogy $\bar{\mathbf{x}}$ optimális megoldás. Az eljárás demonstrálására tekintsük az alábbi 0-1 értékű programozási feladatot:

7.1. példa.

$$\begin{aligned} &5x_1 + 2x_2 + 2x_3 + 2x_4 \leq 5 \\ &x_i \in \{0,1\} \quad (i = 1, \ldots, 4) \\ &\text{---------------------------} \\ &-3x_1 - x_2 - x_3 - x_4 = z(x_1, \ldots, x_4) \to \min \end{aligned}$$

Ekkor

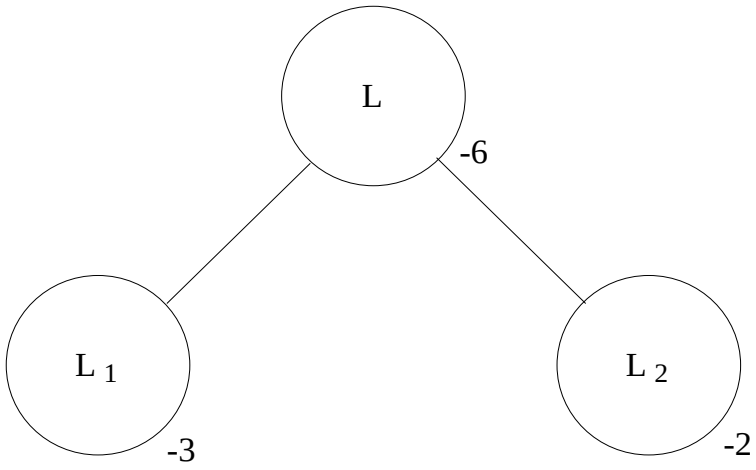
$$L = \{(1,0,0,0,), (0,1,1,0), (0,1,0,1), (0,0,1,1), (0,1,0,0), (0,0,1,0),$$
$$(0,0,0,1), (0,0,0,0)\}.$$

L-hez alsó korlátként hozzárendelhetjük a célfüggvényegyütthatók összegét, azaz tegyük fel, hogy $g(L) = -6$. Tegyük fel továbbá, hogy a φ szétválasztási függvény két részhalmazra bontja L-t, L_1-be kerülnek azok a lehetséges megoldások, amelyeknek első koordinátája 1, L_2-be kerülnek azok a lehetséges megoldások, amelyeknek első koordinátája 0. Akkor

$$L_1 = \{(1,0,0,0,)\} \text{ és } g(L_1) = -3,$$
$$L_2 = \{(0,1,1,0), (0,1,0,1), (0,0,1,1,), (0,1,0,0), (0,0,1,0),$$
$$(0,0,0,1), (0,0,0,0)\}.$$

L_2-nek csak olyan lehetséges megoldások elemei, amelyekben $x_1 = 0$ és az x_2, x_3, x_4 változók közül legfeljebb kettő vehet fel 0-tól különböző értéket, ugyanis ellenkező esetben nem teljesülne az $5x_1+2x_2+2x_3+2x_4 \leq 5$ feltétel. Ezért $z(\mathbf{x}) \geq -2$ teljesül bármely $\mathbf{x} \in L_2$ vektorra.

Tegyük fel, hogy $g(L_2) = -2$. Akkor az alábbi $B\&B$ fához jutunk.



7.2. ábra. A 7.1. példához tartozó $B\&B$ fa.

Mivel L_1 minimális címkéjű, és $L_1 = \{(1,0,0,0)\}$, ezért vége az eljárásnak; $\bar{\mathbf{x}} = (1,0,0,0)$ optimális megoldása a tekintett feladatnak.

Az eljárás ismertetett változatának alkalmazása nehézkes. A megoldandó problémák többségénél L-ről kevés információ áll rendelkezésre és nagyon nehéz az előírt tulajdonságokkal rendelkező szétválasztási függvényt találni. Például, ha az előzőekben megoldott feladatot úgy változtatjuk meg, hogy

a feltételrendszerben x_1 együtthatója 6, akkor az alkalmazott szétválasztási függvényünk már nem lesz jó, $L_1 = \emptyset$ és $L_2 = L$ felbontást eredményez.

A probléma egy lehetséges kiküszöbölése az, hogy L-hez keresünk egy olyan véges Ω halmazt, amely egyrészt tartalmazza L-t, másrészt viszonylag könnyű Ω-hoz szétválasztási függvényt meghatározni. Így a $B\&B$ fák levelei Ω osztályozásainak egy finomodó sorozatát alkotják, és mivel $L \subseteq \Omega$, egyidejűleg implicit módon adódik L osztályozásainak is egy tágabb értelemben vett finomodó sorozata. Nyilvánvaló ahhoz, hogy az eljárás a kívánt módon működjön, a korlátozó függvényre vonatkozó előírásokat alkalmasan meg kell változtatni.

A vizsgált példában $\Omega = \{(u_1, \ldots, u_4) : u_i \in \{0, 1\},\ i = 1, \ldots, 4\}$ tartalmazza L-et, és Ω bármely legalább kételemű részhalmaza szétbontható két nemüres részhalmazra az $x_1, \ldots, x_4$ változók valamelyikének értékei szerint.

Sok esetben heurisztikus eljárásokkal elő lehet állítani viszonylag jó célfüggvényértékkel rendelkező lehetséges megoldásokat. Bizonyos feladatoknál a módszer végrehajtása során részeredményként is előállnak lehetséges megoldások. A rendelkezésre álló lehetséges megoldások felhasználásával esetenként hatékonyabbá tehető az eljárás a következők szerint. Jelölje $\bar{X}$ a rendelkezésre álló lehetséges megoldások halmazát. Legyen $\bar{z} = \min\{z(\mathbf{x}) : \mathbf{x} \in \bar{X}\}$ és jelölje $\mathbf{x}^*$ az $\bar{X}$ halmaz egy olyan elemét, amelyre $z(\mathbf{x}^*) = \bar{z}$ teljesül. Mivel $\bar{X} \subseteq L$, ezért $\bar{z}$ az optimum értékének egy felső korlátja. Most jelölje L' az aktuális $B\&B$ fa valamely leveléhez tartozó halmazt. Ekkor $g(L')$ az L' halmazban levő lehetséges megoldásokon felvett célfüggvényértékek alsó korlátja. Ha $g(L') \geq \bar{z}$, akkor az L' halmazba eső bármely $\tilde{\mathbf{x}}$ lehetséges megoldásra $z(\tilde{\mathbf{x}}) \geq g(L') \geq \bar{z}$ teljesül. Ez pontosan azt jelenti, hogy a tekintett levélhez tartozó lehetséges megoldások között nincs jobb, mint $\mathbf{x}^*$. Viszont ez esetben az aktuális $B\&B$ fa ezen levelének további vizsgálata az optimum meghatározását illetően felesleges, az eljárás folytatása során az illető levéltől el lehet tekinteni. Ilyen esetben szokásos szóhasználat, hogy a $B\&B$ fa megfelelő *ágát lezárjuk.* Ezt a továbbiakban az illető szögpont aláhúzásával jelöljük. Az aktuális fa azon leveleit amelyek nincsenek lezárva *élő leveleknek* nevezzük.

A $B\&B$ fa felépítése igen szemléletessé teszi az eljárást, ezért a továbbiakban mi ezt az ábrázolástechnikát fogjuk használni. Számítástechnikai szempontból viszont nem kívánatos a felesleges információk (lezárt ágak) tárolása. Ez úgy küszöbölhető ki, hogy minden iterációs lépésben csak az aktuális $B\&B$ fa élő leveleit és a hozzájuk tartozó információkat tároljuk.

Az eljárás különböző alkalmazásai során kiderült, hogy az aktuális $B\&B$ fa azon levelének a kiválasztási stratégiája, amelyből a fát továbbépítjük

igen jelentős. Az ismertetett változatban alkalmazott módszer – egy minimális korláttal rendelkező levél kiválasztása – nem minden esetben célszerű. Bizonyos problémacsoportoknál más stratégiák növelik az eljárás hatékonyságát. Így a különböző alkalmazásoknál más és más kiválasztási szabály alapján történhet a $B\&B$ fa kiépítése.

A fentiekben vázolt bővítéseket ötvözi magába a (7.1) probléma megoldására szolgáló, az alábbiakban felépítésre kerülő eljárás.

Tegyük fel, hogy adott egy olyan Ω véges halmaz, amelyre $L \subseteq \Omega$. Továbbá adott egy olyan φ szétválasztási függvény, amely Ω tetszőleges legalább kételemű Ω' részhalmazához hozzárendeli Ω' egy valódi osztályozását. Végül adott egy olyan g korlátozó függvény, hogy Ω tetszőleges nemüres Ω' részhalmazára az alábbiak teljesülnek:

$$g(\Omega') \leq \min\{z(\mathbf{x}) : \mathbf{x} \in \Omega' \cap L\}, \text{ha } |\Omega'| > 1 \text{ és } \Omega' \cap L \neq \emptyset,$$
$$g(\Omega') \text{ tetszőleges, ha } |\Omega'| > 1 \text{ és } \Omega' \cap L = \emptyset,$$

$$g(\Omega') = \begin{cases} z(\bar{\mathbf{x}}), & \text{ha } |\Omega'| = 1 \ \ \& \ \ \Omega' \cap L = \{\bar{\mathbf{x}}\}, \\ W, & \text{ha } |\Omega'| = 1 \ \ \& \ \ \Omega' \cap L = \emptyset, \end{cases}$$

ahol W alkalmasan nagy számot jelöl, például számítógépes implementációban a gépben ábrázolható legnagyobb számot. Esetenként W-hez hozzáadunk vagy levonunk konstansokat. Ezen műveletek eredményét úgy definiáljuk, hogy az ismét W. A g függvénnyel kapcsolatban feltételezzük, hogy az Ω' halmazok (esetleg implicit) leírása olyan, hogy az $|\Omega'| = 1$, $\Omega' \cap L = \{\bar{\mathbf{x}}\}$ eset felismerhető, és $\bar{\mathbf{x}}$ explicit módon előállítható.

Tegyük fel továbbá, hogy ismertek olyan eljárások, amelyekkel a $g(\Omega')$ függvényértékek meghatározhatók bármely $\emptyset \neq \Omega' \subseteq \Omega$ esetén és a $\varphi(\Omega')$ függvényértékek meghatározhatók bármely $\Omega' \subseteq \Omega$ és $|\Omega'| > 1$ részhalmazra. Ekkor a (7.1) feladatot megoldhatjuk az alábbi eljárással.

B&B eljárás

Előkészítő rész. Valamilyen heurisztikus módszerrel határozzunk meg egy lehetséges megoldást, jelölje ezt $\mathbf{x}_0$. Legyen a $\bar{z}$ változó értéke $z(\mathbf{x}_0)$ és az $\mathbf{x}^*$ vektorváltozó értéke $\mathbf{x}_0$. Ha lehetséges megoldás meghatározására nincsen lehetőség, akkor legyen $\bar{z} = W$, és $\mathbf{x}^* = (-1, \ldots, -1)$.

Határozzuk meg a $g(\Omega)$ korlátot. Ha a $g(\Omega)$ korlát meghatározása során előáll egy $\bar{\mathbf{x}}$ lehetséges megoldás, akkor definiáljuk újra $\bar{z}$-t és $\mathbf{x}^*$-ot a következők szerint. Ha $z(\bar{\mathbf{x}}) \geq \bar{z}$, akkor $\bar{z}$ és $\mathbf{x}^*$ nem változnak, míg $z(\bar{\mathbf{x}}) < \bar{z}$

esetében $\bar{z}$-nak adjuk a $z(\bar{\mathbf{x}})$ értéket és $\mathbf{x}^*$-ot változtassuk $\bar{\mathbf{x}}$-ra. Legyen $r = 0$ és

$$F_0 = \begin{cases} \{\Omega\}, & \text{ha } g(\Omega) < \bar{z}, \\ \emptyset & \text{különben.} \end{cases}$$

Ezt követően folytassuk az eljárást az iterációs eljárásrésszel.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $F_r = \emptyset$, akkor vége az eljárásnak; $\mathbf{x}^*$ a feladat optimális megoldása, $\bar{z}$ az optimum értéke. Különben folytassuk a 2. lépéssel az eljárást.

- *2. lépés.* Ha $F_r \neq \emptyset$, akkor valamilyen rögzített stratégia szerint válasszunk ki egy elemet F_r elemei közül, jelölje ezt Ω'. Alkalmazzuk a φ szétválasztási függvényt Ω'-re. Legyen

$$\varphi(\Omega') = \{\Omega_1^{(r)}, \ldots, \Omega_{k_r}^{(r)}\}.$$

  Határozzuk meg rendre az $\Omega_1^{(r)}, \ldots, \Omega_{k_r}^{(r)}$ halmazokra a $g(\Omega_1^{(r)}), \ldots, g(\Omega_{k_r}^{(r)})$ korlátokat. Jelölje X_r a korlátok meghatározása során előálló lehetséges megoldások halmazát, feltéve, hogy ilyen létezik. Legyen $z_r = \min\{z(\mathbf{x}) : \mathbf{x} \in X_r\}$ és jelölje $\mathbf{x}^{(r)}$ az X_r halmaz egy olyan elemét, amelyre $z_r = z(\mathbf{x}^{(r)})$ teljesül. Definiáljuk újra $\bar{z}$-t és $\mathbf{x}^*$-ot a következők szerint. Ha $z_r \geq \bar{z}$, akkor $\bar{z}$ és $\mathbf{x}^*$ nem változik, míg $z_r < \bar{z}$ esetén $\bar{z}$-nak adjuk a z_r értéket és $\mathbf{x}^*$-ot változtassuk $\mathbf{x}^{(r)}$-re. ($X_r = \emptyset$ esetén $\bar{z}$ és $\mathbf{x}^*$ nem változnak.). Ezek után legyen

$$F_{r+1} = \{\Omega_i : \Omega_i \in (F_r \setminus \{\Omega'\}) \cup \varphi(\Omega') \ \ \& \ \ g(\Omega_i) < \bar{z}\}.$$

  Növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

Az eljárás helyességének igazolása

Vegyük észre, hogy az eljárás során implicit módon fák egy olyan sorozatát állítjuk elő, hogy minden egyes fa leveleihez Ω egy osztályozása tartozik, és az osztályozásokat minden iterációs lépésben finomítjuk. Az F_r halmazok praktikussági okokból csak mindig az aktuális fa élő leveleihez tartozó halmazokat tartalmazzák. Ezek a halmazok legalább kételeműek,

ugyanis $|\Omega'| = 1$ esetén $g(\Omega') = W$ vagy $g(\Omega') = z(\bar{\mathbf{x}})$ teljesül valamely $\bar{\mathbf{x}} \in L$ lehetséges megoldásra. Az első esetben az Ω'-höz tartozó levél lezárásra kerül, a második esetben pedig a g-re vonatkozó feltételezésünk szerint $\bar{\mathbf{x}}$ rendelkezésünkre áll, de akkor $\bar{\mathbf{x}} \in X_r$ és így F_{r+1} kialakításakor $g(\Omega') = z(\bar{\mathbf{x}}) \geq \bar{z}$. Viszont így az Ω'-höz tartozó levél ismét lezárásra kerül.

A fenti észrevételeket felhasználva, az eljárás helyessége hasonlóan bizonyítható, mint a megelőző verzióé.

Vegyük még észre, hogy a φ szétválasztási és a g korlátozó függvényekre szükségtelenül erős az a kikötés, amely szerint φ tetszőleges $\Omega' \subseteq \Omega$ és $|\Omega'| > 1$ részhalmazra, g tetszőleges $\emptyset \neq \Omega' \subseteq \Omega$ részhalmazra értelmezve van. Ez sok esetben nagyon körülményes, vagy egyáltalán nem biztosítható. Lényegében elegendő, ha a φ szétválasztási függvény értelmezve van az eljárás során előálló $B\&B$ fák élő levelein. A g korlátozó függvényről pedig elegendő azt kikötni, hogy értelmezett az Ω halmazon és az eljárás során előálló $B\&B$ fák bármely Ω' élő levelére $\varphi(\Omega')$ elemein.

A φ és g függvényekre vonatkozó fenti kikötések is formalizálhatók. Mivel az általános eset formalizmusa eléggé bonyolult, ezért ettől eltekintünk. Az egyes modelleknél rendre megadjuk a konkrét φ és g függvényeket és tulajdonságaikat.

A következőkben a fenti eljárást fogjuk alkalmazni konkrét problémacsoportok megoldására. Egy konkrét alkalmazás esetén meg kell adni

(a) *az Ω halmazt,*

(b) *a φ szétválasztási és a g korlátozó függvényt, és igazolni kell, hogy ezek rendelkeznek a kívánt tulajdonságokkal,*

(c) *azt a stratégiát, amely alapján az aktuális $B\&B$ fa levelét kiválasztjuk további faépítés céljából.*

A felsoroltak megadásával egy konkrét eljárást kapunk az illető problémacsoport megoldására, amelynek helyessége következik az általános eljárás helyességéből.

Az eljárással kapcsolatban megjegyezzük még, hogy nyilvánvaló módon maximum feladatok megoldására is alkalmas. Ez esetben W helyett mindenhol $-W$ szerepel, minimumok helyett maximumokkal kell dolgozni, és a célfüggvényértékekre, valamint a korlátokra vonatkozó egyenlőtlenségekben "$\leq$" helyett "$\geq$" relációt, "$<$" helyett "$>$" relációt kell venni, végül a g korlátozó függvénynek az egyes halmazokhoz felső korlátokat kell rendelnie.

8. Hátizsák feladat

A hátizsák feladatnak számos gyakorlati alkalmazása van, ezek közül az egyik legismertebb a következő.

Hátizsák feladat

Adott egy hátizsák és különböző használati tárgyak egy halmaza. Minden egyes tárgynak adott a súlya és az értéke, továbbá a hátizsákba kerülő rakomány súlya is korlátozva van (nem lehet bármilyen nagy súlyú rakományt a hátizsákban elszállítani). Feladat a rendelkezésre álló tárgyakból egy olyan rakományt összeállítani, amely szállítható (a súlya nem haladja meg a rakomány súlykorlátját) és emellett maximális értékű.

Jelölje

m a használati tárgyak számát,
a_j a j-dik tárgy súlyát, $(j = 1, \ldots, m)$,
c_j a j-dik tárgy értékét, $(j = 1, \ldots, m)$,
b a hátizsák rakományának súlykorlátját, továbbá legyen

$$x_j = \begin{cases} 1, & \text{ha a } j\text{-edik tárgy bekerül a rakományba,} \\ 0 & \text{különben.} \end{cases}$$

A bevezetett jelölésekkel a hátizsák feladat a következő optimumszámítási modellel írható le:

$$\begin{gathered} \textstyle\sum_{j=1}^{m} a_j x_j \le b \\ x_j \in \{0,1\},\ (j = 1, \ldots, m) \\ \hline \textstyle\sum_{j=1}^{m} c_j x_j \to \max \end{gathered} \tag{8.1}$$

Nyilvánvalóan az első feltétel baloldala a rakomány súlyát adja meg, ami nem haladhatja meg a jobboldalon álló b felső korlátot. A célfüggvény pedig a rakomány értékét adja meg. A második feltételcsoport azt írja le, hogy különbözők a tárgyak, és minden egyes tárgy vagy benne lesz a rakományba vagy nem lesz benne.

A (8.1) feladatot szokás *bináris hátizsák feladatnak* vagy $0-1$ *hátizsák feladatnak* nevezni abból a célból, hogy megkülönböztessük más típusú hátizsák feladatoktól.

Az egyik természetes általánosítása a bináris hátizsák feladatnak, az a feladat, amikor a rendelkezésre álló tárgyak több példányban, esetleg korlátlan mennyiségű példányban használhatók fel. Ilyenkor az x_j változó azt adja meg, hogy a j-edik tárgyból hány példány kerül a rakományba. A megfelelő optimumszámítási modell a következő:

$$\begin{array}{c} \sum_{j=1}^{m} a_j x_j \leq b \\ x_j \geq 0 \text{ és } x_j \text{ egész } (j = 1, \ldots, m) \\ \hline \sum_{j=1}^{m} c_j x_j \to \max \end{array} \tag{8.2}$$

A (8.2) modellt szokás *egészértékű hátizsák feladatnak* nevezni. Ezen belül is megkülönböztethető két fajta, a *korlátos* változat, amikoris minden x_j változóra a feladatban adott egy felső korlát és a nemkorlátos változat, amikor nincs minden x_j-re felső korlát. Ha egyetlen x_j változóra sincs felső korlát, akkor *általános hátizsák feladatról* beszélünk.

Mielőtt bármit kezdenénk a hátizsák feladattal, tegyük fel, hogy a továbbiakban csak olyan (8.1) és (8.2) típusú feladatokkal foglalkozunk, amelyekre

(1) a_j egész, $(j = 1, \ldots, m)$,

(2) c_j egész, $(j = 1, \ldots, m)$,

(3) b egész.

A (8.2) modell egy gyakorlati alkalmazása a következő feladat.

Pénzváltási feladat

Adottak bizonyos fajta pénzérmék. Mindegyik fajtának meghatározott értéke van és korlátlan mennyiségben állnak rendelkezésre. Másrészt adott egy pénzmennyiség, amelyet pénzérmékre akarunk átváltani úgy, hogy az ellenértékül kapott pénzérmék száma minimális legyen.

Legyen a pénzérme fajták száma m. Jelölje a j-edik fajta pénzérme értékét a_j. Továbbá legyen b az a pénzmennyiség, amit érmékre akarunk váltani. Végül legyen x_j a j-edik érmetípusból felhasznált érmék száma. Akkor a pénzváltási feladat olyan (8.2) típusú feladattal írható le, amelyben az első feltételnél egyenlőséget követelünk meg és $c_j = -1$, $(j = 1, \ldots, m)$. Nevezzük *általánosított pénzváltási feladatnak* azt a (8.2) típusú feladatot, amelyben a feltétel egyenlőség.

Fontos megjegyezni, hogy a hátizsák feladat önmagában is érdekes, de ezen kívül számos bonyolultabb típusú feladat vizsgálataiban, megoldó eljárásaiban is kulcsszerepet játszik. Többek között ez annak az oka, hogy nagyon sok eredmény jelent meg a hátizsák feladatokról, melyekről jó ismertetések és összefoglalók találhatók a [132], [158], [168] munkákban.

Az elmondottak alapján nyilvánvaló, hogy a jelen fejezet kereteiben csak arra van lehetőség, hogy válogassunk a sok eredményből. Ezt is tettük, és a válogatás eredményeként elsőként megmutatjuk, hogy számos egészértékű lineáris programozási feladat visszavezethető egy alkalmas általánosított pénzváltási feladat megoldására. Ezt követően bináris hátizsák feladatokra építünk fel megoldó eljárásokat, melyek közül az első a dinamikus programozás elvére épül, a második pedig egy Branch-and-Bound eljárás.

Tudjuk, hogy az i-edik egyenlőtlenség baloldalát bővítve egy nemnegatív egészértékű u_i eltérésváltozóval ($i = 1, \dots n$), az $\mathbf{Ax} \leq \mathbf{b}$ feltételcsoportról áttérhetünk olyan feltételrendszerre, amelyben már csak egyenlőségek szerepelnek. Ezért rögtön ilyen egészértékű lineáris programozási feladatot tekintünk. Nevezetesen tekintsük az

$$\mathbf{A}^{n\times m}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0} \text{ és } \mathbf{x} \text{ egész}$$
$$\overline{\qquad\qquad\qquad\qquad}$$
$$\mathbf{cx} = z \to \max$$

egészértékű lineáris programozási feladatot, ahol $\mathbf{A}$, $\mathbf{b}$ és $\mathbf{c}$ rendre egészek. Jelölje a fenti feladat lehetséges megoldásainak a halmazát L^*.

Megmutatjuk, hogy bizonyos feltételek mellett a tekintett lineáris programozási feladat visszavezethető egy alkalmas általánosított pénzváltási feladatra. A visszavezetés alapötlete az, hogy az

$$\mathbf{A}^{n\times m}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0} \text{ és } \mathbf{x} \text{ egész}$$

feltételrendszerhez meghatározunk egy olyan $\mathbf{w} = (w_1, \dots, w_n)$ vektort, amelyre a

$$\mathbf{wAx} = \mathbf{wb},\ \mathbf{x} \geq \mathbf{0} \text{ és } \mathbf{x} \text{ egész}$$

új feltételrendszert (ami egyetlen egyenlet) pontosan azok az $\mathbf{x}$ vektorok elégítsék ki, amelyek kielégítik az előző feltételrendszert. Jelölje $L^*_{\mathbf{w}}$ az új feltételrendszer által meghatározott lehetséges megoldások halmazát. Olyan $\mathbf{w}$-re van szükségünk, amelyre

$$L^* = L^*_{\mathbf{w}}$$

Nyilvánvalóan elegendő $n = 2$-re megoldást adni, akkor kettesével össze lehet vonni az egyenleteket, először az első kettőt, majd az összevont egyenlettel

a harmadikat, és így tovább. Ezt a technikát szokás *aggregációnak* nevezni. Az első aggregációs eljárás G. Mathewstől [140] származik. Mi egy másik megoldással fogunk megismerkedni, amely F. Glovertől [75] származik 1972-ből.

8.1. tétel ([75]). *Legyen a tekintett két egyenlet* $s_i(\mathbf{x}) = \sum_{j=1}^{m} a_{ij}x_j = b_i$, $(i = 1, 2)$, *ahol* b_1 *és* b_2 *valamelyike nem zéró. Ha* w_1, w_2 *olyan nem 0, relatív prímek, hogy*

(1) $w_1b_1 + w_2b_2 \neq 0$,

(2) $w_1a_{1j} + w_2a_{2j} \geq |b_2a_{1j} - b_1a_{2j}|$, $(j = 1, \ldots, m)$,

(3) $(\forall \bar{\mathbf{x}} \in L^*_{\mathbf{w}})(\exists j \in \{1, \ldots, m\})(\ w_1a_{1j} + w_2a_{2j} > |b_2a_{1j} - b_1a_{2j}|$ *és* $\bar{x}_j > 0)$,

akkor $L^* = L^*_{\mathbf{w}}$ *feltéve, hogy* $L^* \neq \emptyset$.

Bizonyítás. $L^* \subseteq L^*_{\mathbf{w}}$ triviális, ugyanis, ha egy nemnegatív egész $\bar{\mathbf{x}}$ vektor kielégíti a két egyenletet, akkor kielégíti azok bármilyen lineáris kombinációját is.

Most megmutatjuk, hogy $L^*_{\mathbf{w}} \subseteq L^*$ szintén teljesül. Feltevésünk miatt $L^* \neq \emptyset$, továbbá az előzőek szerint $L^* \subseteq L^*_{\mathbf{w}}$. Ezzel $L^*_{\mathbf{w}} \neq \emptyset$. Legyen $\bar{\mathbf{x}} \in L^*_{\mathbf{w}}$. Akkor $\bar{\mathbf{x}} \neq \mathbf{0}$, mivel $\bar{\mathbf{x}}$ kielégíti a $\mathbf{wAx} = \mathbf{wb}$ egyenletet. Ha $\bar{\mathbf{x}} = \mathbf{0}$ lenne, akkor az egyenlet baloldalán 0-át kapnánk, míg a jobboldal $w_1b_1 + w_2b_2$, és ez utóbbi érték a feltétel szerint nem nulla. Szintén a feltétel szerint

$$w_1a_{1j} + w_2a_{2j} \geq |b_2a_{1j} - b_1a_{2j}|, \ (j = 1, \ldots, m),$$

és van olyan $j_0 \in \{1, \ldots, m\}$, hogy

$$w_1a_{1j_0} + w_2a_{2j_0} > |b_2a_{1j_0} - b_1a_{2j_0}|, \ \bar{x}_{j_0} > 0.$$

Megszorozva mindkét oldalt az $\bar{x}_j \geq 0$-val és összegezve j-re,

$$\sum_{j=1}^{m}(w_1a_{1j} + w_2a_{2j})\bar{x}_j > \sum_{j=1}^{m} |b_2a_{1j} - b_1a_{2j}|\bar{x}_j$$

Rendezve az egyenlőtlenséget,

$$w_1 \sum_{j=1}^{m} a_{1j}\bar{x}_j + w_2 \sum_{j=1}^{m} a_{2j}\bar{x}_j > |b_2 \sum_{j=1}^{m} a_{1j}\bar{x}_j - b_1 \sum_{j=1}^{m} a_{2j}\bar{x}_j|,$$

amit az $s_1(\mathbf{x})$, $s_2(\mathbf{x})$ jelölésekkel az alábbi alakban írhatunk fel:

$$w_1s_1(\bar{\mathbf{x}}) + w_2s_2(\bar{\mathbf{x}}) > |b_2s_1(\bar{\mathbf{x}}) - b_1s_2(\bar{\mathbf{x}})|.$$

Mivel $\bar{\mathbf{x}} \in L^*_{\mathbf{w}}$, ezért a baloldalra

$$w_1s_1(\bar{\mathbf{x}}) + w_2s_2(\bar{\mathbf{x}}) = w_1b_1 + w_2b_2.$$

Most a $w_1b_1 + w_2b_2$ kifejezést behelyettesítve, azt kapjuk, hogy

$$(*) \qquad w_1b_1 + w_2b_2 > |b_2s_1(\bar{\mathbf{x}}) - b_1s_2(\bar{\mathbf{x}})|.$$

Most indirekt igazoljuk, hogy $\bar{\mathbf{x}} \in L^*$. Tegyük fel, hogy $\bar{\mathbf{x}} \notin L^*$. Akkor az $s_1(\mathbf{x}) = b_1$ és $s_2(\mathbf{x}) = b_2$ egyenletek valamelyikét $\bar{\mathbf{x}}$ nem elégíti ki. Tegyük fel, hogy $s_1(\bar{\mathbf{x}}) \neq b_1$. Akkor $\bar{\mathbf{x}} \in L^*_{\mathbf{w}}$-ből következik, hogy

$$w_1 s_1(\bar{\mathbf{x}}) + w_2 s_2(\bar{\mathbf{x}}) = w_1 b_1 + w_2 b_2.$$

Átrendezve az egyenletet, azt kapjuk, hogy $w_1(s_1(\bar{\mathbf{x}}) - b_1) = w_2(b_2 - s_2(\bar{\mathbf{x}}))$. Mivel $w_1 \neq 0$ és $w_2 \neq 0$, ezért a kapott egyenlőségből az következik, hogy

$$s_1(\bar{\mathbf{x}}) \neq b_1 \Longleftrightarrow s_2(\bar{\mathbf{x}}) \neq b_2$$

Tehát léteznek olyan nemzéró α, β egészek, hogy

$$s_1(\bar{\mathbf{x}}) = b_1 + \alpha \quad \text{és} \quad s_2(\bar{\mathbf{x}}) = b_2 - \beta.$$

Behelyettesítve $\bar{\mathbf{x}}$-t az új feladat alábbi feltételébe

$$w_1 s_1(\mathbf{x}) + w_2 s_2(\mathbf{x}) = w_1 b_1 + w_2 b_2,$$

$\bar{\mathbf{x}}$ kielégíti ezt a feltételt, így azt kapjuk, hogy

$$w_1(b_1 + \alpha) + w_2(b_2 - \beta) = w_1 b_1 + w_2 b_2, \text{ amiből } w_1\alpha = w_2\beta.$$

Mivel w_1 és w_2 relatív prímek és minden tényező nem nulla, ezért létezik olyan $q \neq 0$ egész, hogy $\alpha = q w_2$. Kifejezve β-t is: $\beta = w_1\alpha / w_2 = w_1 q w_2 / w_2 = q w_1$. Ekkor

$$s_1(\bar{\mathbf{x}}) = b_1 + q w_2 \quad \text{és} \quad s_2(\bar{\mathbf{x}}) = b_2 - q w_1.$$

Most az $s_1(\bar{\mathbf{x}})$-re és az $s_2(\bar{\mathbf{x}})$-re kapott kifejezéseket behelyettesítve a (*) egyenlőtlenségbe

$$w_1 b_1 + w_2 b_2 > |b_2(b_1 + q w_2) - b_1(b_2 - q w_1)|$$

amiből

$$w_1 b_1 + w_2 b_2 > |q(w_1 b_1 + w_2 b_2)|.$$

Vegyük észre, hogy az utolsó egyenlőtlenség csak akkor teljesülhet, ha $q = 0$, mivel q egész és $w_1 b_1 + w_2 b_2 \neq 0$, amivel ellentmondáshoz jutottunk. Tehát $s_1(\bar{\mathbf{x}}) = b_1$ és $s_2(\bar{\mathbf{x}}) = b_2$, azaz $\bar{\mathbf{x}} \in L^*$, amivel a 8.1. tételt igazoltuk.

A továbbikban a bináris esetre egy példán megmutatjuk, hogy miként lehet a megfelelő $\mathbf{w}$ vektort meghatározni, és vele az új feltételt előállítani.

8.1. példa.

$$\begin{gathered} 2x_1 - 3x_2 + 4x_3 \leq 5 \\ -3x_1 - x_2 + 2x_3 \leq -1 \\ x_j \in \{0,1\}, \quad (j = 1,2,3) \\ \hline -2x_1 - x_2 - 3x_3 = z \to \max \end{gathered}$$

Egyszerűen ellenőrizhető, hogy a lehetséges megoldások halmaza a következő:

$$\bar{L} = \{(0,1,0),(1,0,0),(1,1,0),(1,1,1)\}.$$

Kiegészítve az egyenlőtlenségek baloldalát az x_5 és x_6 eltérésváltozókkal, a kapott feltételrendszer lehetséges megoldásainak L^* halmaza:

$$L^* = \{(0,1,0,8,0),(1,0,0,3,2),(1,1,0,6,3),(1,1,1,2,1)\}.$$

A w_1 és w_2 konstansok könnyebb meghatározásához, ha valamely feltételben x_j együtthatója negatív, akkor ott x_j-t helyettesítjük $(1-x'_j)$-vel. Most végrehajtva a negatív együtthatóknál a helyettesítéseket és feltüntetve a két egész eltérésváltozót, a következő feladatot kapjuk:

$$\begin{array}{r} 2x_1 + \quad +3x'_2 + 4x_3 + x_4 \quad = 8 \\ 3x'_1 + x'_2 + 2x_3 \quad + x_5 = 3 \\ \hline -2x_1 - x_2 - 3x_3 = z \to \max \end{array}$$

Az új egyenletek baloldalait jelölje $s'_1(\mathbf{x})$ és $s'_2(\mathbf{x})$. Ekkor az egyenletekben szereplő változókat és azok együtthatóit adja meg az alábbi táblázat.

	x_1	x'_1	x'_2	x_3	x_4	x_5	b_i
s'_1	2		3	4	1		8
s'_2		3	1	2		1	3

8.1. táblázat. A változók és együtthatóik.

Most olyan w_1 és w_2 értékeket kell meghatároznunk, hogy teljesüljenek a 8.1. tétel feltételei. Ehhez nyújt segítséget az alábbi táblázat, ahol minden változóra megadjuk a (2) feltétel jobboldalán szereplő értéket, valamint w_1, w_2-re egy olyan relációt, ami biztosítja, hogy a (2) feltétel teljesüljön.

j	$\lvert b_2a_{1j} - b_1a_{2j}\rvert$	$w_1a_{1j} + w_2a_{2j}$
1	$3\cdot 2 + 8\cdot 0 = 6$	$w_1\cdot 2 > 6 \to w_1 > 3$
$1'$	$3\cdot 0 + 8\cdot 3 = 24$	$w_2\cdot 3 > 24 \to w_2 > 8$
$2'$	$3\cdot 3 - 8\cdot 1 = 1$	$w_1\cdot 3 + w_2\cdot 1 > 1 \to w_j > 1$
3	$\lvert 3\cdot 4 - 8\cdot 2\rvert = 4$	$w_1\cdot 4 + w_2\cdot 2 > 4 \to w_j > 2$
4	$3\cdot 1 - 8\cdot 0 = 3$	$w_1\cdot 1 > 3 \to w_1 > 3$
5	$\lvert 3\cdot 0 - 8\cdot \rvert = 8$	$w_2\cdot 1 > 8$

A táblázatból azt kapjuk, hogy w_1 alsó korlátjai a $3, 1, 2, 3$ egészek, míg w_2 alsó korlátjai a $8, 1, 2, 8$ egészek. Ahhoz, hogy a 8.1. tétel minden feltétele fennálljon, olyan w_1, w_2 egészeket kell választani, hogy w_1 és w_2 relatív prímek legyenek, $w_1b_1 + w_2b_2 \neq 0$, valamint $w_1 > 3$, $w_2 > 8$ teljesüljenek. Könnyen ellenőrizhető, hogy $w_1 = 4$, $w_2 = 9$ esetében a felsorolt feltételek teljesülnek. Most nézzük rendre a 8.1. tétel feltételeit.

A tekintett egyenletrendszerre $b_1 = 8$ és $b_2 = 3$, azaz egyikük nemzéró.

Másrészt $w_1 \neq 0$ és $w_2 \neq 0$ olyan relatív prímek, hogy

(1) $w_1b_1 + w_2b_2 \neq 0$.

(2) $w_1a'_{1j} + w_2a'_{2j} \geq |b_2a'_{1j} - b_1a'_{2j}|$, $(j = 1, \ldots, m)$,

(3) $(\forall \bar{\mathbf{x}} \in L^*_{\mathbf{w}})(\exists j \in \{1, \ldots, m\})(\ w_1a'_{1j} + w_2a'_{2j} > |b_2a'_{1j} - b_1a'_{2j}|$ és $\bar{x}_j > 0)$,

(4) $L^* \neq \emptyset$.

A (2) feltétel a w_1 és w_2 választása miatt teljesül, ráadásul minden változóra a $>$ reláció érvényes. (Itt az a'_{ij} jelöléssel arra utaltunk, hogy ezek már nem az eredeti együtthatók, hanem az x'_j változók bevezetése és a különbségváltozók felvétele utáni együtthatók.

A (3) feltétel esetében vegyük észre azt, hogy az új feladat (ami egyetlen egyenletet tartalmaz) jobboldala $w_1b_1 + w_2b_2$, ami (1) alapján nem 0. De akkor $\mathbf{0} \notin L^*_{\mathbf{w}}$, azaz minden $\bar{\mathbf{x}} \in L^*_{\mathbf{w}}$-re van olyan j, hogy $\bar{x}_j > 0$. De ezzel a (3) feltétel már teljesül, mivel (2)-ben a $>$ reláció érvényes minden változóra, így a tekintett x_j-re is.

L^*-ról tudjuk, hogy 4 elemet tartalmaz, így $L^* \neq \emptyset$.

Következésképpen a 8.1. tétel minden feltétele teljesül, de akkor a 8.1. tétel alapján $L^* = L^*_{\mathbf{w}}$.

Most az alkalmas w_1 és w_2 ismeretében határozzuk meg a konkrét új egyenletet.

$$\begin{aligned} s'_1(\mathbf{x}) &= 2x_1 + \quad +3x'_2 + 4x_3 + x_4 \quad = 8 \\ s'_2(\mathbf{x}) &= \quad 3x'_1 + x'_2 + 2x_3 \quad + x_5 = 3 \end{aligned}$$

$$w_1 = 4,\ w_2 = 9.$$

Behelyettesítve a $w_1s'_1(\mathbf{x}) + w_2s'_2(\mathbf{x}) = w_1b_1 + w_2b_2$ egyenletbe,

$$8x_1 + 27x'_1 + 21x'_2 + 34x_3 + 4x_4 + 9x_5 = 59.$$

Helyettesítve x'_j-t $(1 - x_j)$-vel,

$$8x_1 - 27x_1 - 21x_2 + 34x_3 + 4x_4 + 9x_5 = 11, \text{ azaz}$$

$$-19x_1 - 21x_2 + 34x_3 + 4x_4 + 9x_5 = 11$$

Most meghatározzuk a kapott egyenlet lehetséges megoldásait. Ezt például a következő módon tehetjük meg. Rendre értéket adva az x_1,x_2,x_3 bináris változóknak, egy $4x_4+9x_5 = \alpha$ alakú diofantoszi egyenlethez jutunk, amelynek nemnegatív megoldásai a bináris értékekkel kiegészítve, adják az $L^*_{\mathbf{w}}$-ben levő lehetséges megoldásokat. Ismeretes (ld. pl. [148]), hogy a $4x_4 + 9x_5 = \alpha$ egyenletnek létezik megoldása, mivel 4 és 9 relatív prímek. Sőt azt is tudjuk, hogy ilyenkor végtelen sok megoldás van, amelyek $(x_0 + 9t, y_0 - 4t)$ alakúak, ahol (x_0, y_0) egy megoldása a $4x_4+9x_5 = \alpha$ egyenletnek. Az egyszerűbb érthetőség kedvéért újra felírjuk a vizsgálat tárgyát képező egyenletet:

$$-19x_1 - 21x_2 + 34x_3 + 4x_4 + 9x_5 = 11.$$

Vizsgáljuk az $x_1 = 0$, $x_2 = 0$, $x_3 = 0$ esetet. Ekkor a diofantoszi egyenlet: $4x_4 + 9x_5 = 11$, amelynek nem létezik nemnegatív egész megoldása. Tehát $L^*_{\mathbf{w}}$ nem tartalmaz olyan vektort, amelynek az első három komponense 0-val egyenlő.

Ha $x_1 = 0, x_2 = 0, x_3 = 1$, akkor a diofantoszi egyenlet: $4x_4+9x_5 = -23$, amelynek nincsen nemnegatív megoldása. Tehát $L^*_{\mathbf{w}}$ nem tartalmaz olyan vektort, amelynek az első három komponense $0, 0, 1$.

Legyen $x_1 = 0$, $x_2 = 1$, $x_3 = 0$. Ekkor a diofantoszi egyenlet: $4x_4+9x_5 = 32$, amelynek pontosan egy nemnegatív megoldása van, az $x_4 = 8$, $x_5 = 0$. Tehát $(0, 1, 0, 8, 0) \in L^*_{\mathbf{w}}$, és $L^*_{\mathbf{w}}$-ben nincsen más olyan vektor, amelynek első három komponense $0, 1, 0$.

Ha $x_1 = 0, x_2 = 1, x_3 = 1$, akkor a diofantoszi egyenlet: $4x_4 + 9x_5 = -2$, amelynek nincs nemnegatív megoldása, így $L^*_{\mathbf{w}}$ nem tartalmaz olyan vektort, amelynek az első három komponense $0, 1, 1$.

Ha $x_1 = 1$, $x_2 = x_3 = 0$, akkor a diofantoszi egyenlet: $4x_4 + 9x_5 = 30$, aminek pontosan egy nemnegatív megoldása van $x_4 = 3$, $x_5 = 2$. Így $(1, 0, 0, 3, 2) \in L^*_{\mathbf{w}}$ és $L^*_{\mathbf{w}}$ nem tartalmaz más olyan vektort, amelynek az első három komponense $1, 0, 0$.

Hasonlóan folytatva a vizsgálatokat, pontosan megkapjuk azokat a vektorokat, amelyek az L^* halmaz elemei.

A fejezet további részeiben már csak bináris hátizsák feladatokkal fogunk foglalkozni, nevezetesen azt vizsgáljuk, hogy miként lehet ezeket megoldani. Ismeretes, hogy a bináris hátizsák probléma NP-nehéz (ld. pl. [68], [132]), ezért nem várható hatékony megoldó eljárás ezekre a feladatokra. Elsőként megmutatjuk, hogy elegendő csak bizonyos típusú feladatok megoldására

szorítkozni, aztán ezt felhasználva, megadunk két megoldó eljárást, melyek végrehajtását példákon fogjuk demonstrálni. Tekintsük ezek után a (8.1) bináris hátizsák feladatot.

$$\begin{gathered}\sum_{j=1}^{m} a_j x_j \le b \\ x_j \in \{0,1\},\ (j=1,\dots,m) \\ \hline \sum_{j=1}^{m} c_j x_j \to \max\end{gathered}$$

8.1. segédtétel. *Az optimális megoldás létezését és meghatározását illetően elegendő olyan (8.1) típusú feladatok vizsgálatára szorítkozni, amelyekben minden a_j, c_j együttható pozitív egész.*

Bizonyítás. Megadunk egy olyan eljárást, amely tetszőleges hátizsák feladathoz konstruál egy olyan új hátizsák feladatot, hogy a két feladatnak egyidejűleg létezik optimális megoldása, és az új feladat optimális megoldásából közvetlenül származtatható a kiindulási feladat optimális megoldása.

- *1. lépés.* Minden olyan $j \in \{1,\dots,m\}$ indexre, amelyre $c_j \le 0$ és $a_j \le 0$, helyettesítsük x_j-t $(1 - x'_j)$-vel.
- *2. lépés.* Minden $j \in \{1,\dots,m\}$ indexre, ha $c_j > 0$ és $a_j \le 0$, akkor adjuk az x_j változónak az 1 értéket, és rendezzük át a feladatot.
- *3. lépés.* Minden $j \in \{1,\dots,m\}$-re, ha $c_j \le 0$ és $a_j > 0$, akkor adjuk az x_j változónak a 0 értéket.

Az eljárás helyessége egyszerűen igazolható, ezért ezt most nem bizonyítjuk.

Vegyük még észre, hogy amennyiben minden a_j, c_j pozitív, akkor b-nek is szükségképp pozitívnak kell lennie, ugyanis ellenkező esetben triviális, hogy a feladatnak nincs lehetséges megoldása, vagy csak egy lehetséges megoldása van, a $\mathbf{0}$ vektor.

Az átalakítás illusztrálására tekintsük a következő feladatot, amelynél a rövidség kedvéért a binaritási feltételeket nem tüntettük fel.

8.2. példa.

$$\begin{gathered}x_2 + 2x_3 + 3x_4 - x_5 - 2x_6 - 2x_7 \le 4 \\ \hline x_1 \quad + 3x_3 \ - x_4 - 2x_5 + x_6 \ - x_7 = z \to \max\end{gathered}$$

Az 1. lépés szerint helyettesítsük x_5-öt $1 - x'_5$-vel és x_7-et $1 - x'_7$-vel. A kapott új hátizsák feladat a következő:

$$\frac{x_2 + 2x_3 + 3x_4 + x'_5 - 2x_6 + 2x'_7 \leq 7}{-3 + x_1 + 3x_3 \ - x_4 + 2x'_5 + x_6 \ + x'_7 = z \to \max}$$

Most a 2. lépés szerint adjuk az x_1 és x_6 változóknak az 1 értéket. A helyettesítés utáni rendezéssel az alábbi hátizsák feladathoz jutunk.

$$\frac{x_2 + 2x_3 + 3x_4 + x'_5 + 2x'_7 \leq 9}{-1 + 3x_3 \ - x_4 + 2x'_5 \ + x'_7 = z \to \max}$$

Most a 3. lépés szerint az x_2 és x_4 változóknak 0 értéket adva, a következő hátizsák feladatot kapjuk, amelyben már minden együttható pozitív.

$$\frac{2x_3 + x'_5 + 2x'_7 \leq 9}{-1 + 3x_3 \ + 2x'_5 \ + x'_7 = z \to \max}$$

A továbbiakban olyan feladatok megoldására adunk megoldó eljárást, amelyekben minden együttható pozitív. Az első tárgyalásra kerülő eljárásunk a dinamikus programozás elvén alapul. Ennek tárgyalásába a dinamikus programozás módszerét nem vesszük be, hanem direkt módon mutatjuk meg az eljárás helyességét. A dinamikus programozást és a hátizsák feladatot összekapcsoló munkák iránt érdelődőknek ajánljuk a [14], [15], [16], [17], [40] úttörő jellegű munkákat.

Dinamikus programozáson alapuló megoldó eljárás

Legyen a tekintett feladat

$$\frac{\sum_{j=1}^{m} a_j x_j \leq b \quad x_j \in \{0,1\},\ (j = 1, \ldots, m)}{\sum_{j=1}^{m} c_j x_j = z \to \max}$$

amelyben minden együttható pozitív egész. Vegyük észre, hogy ennek a feladatnak mindig létezik optimális megoldása, mivel $\mathbf{0} \in L$.

Minden $k \in \{1, \ldots, m\}$ és $r \in \{0, 1, \ldots, b\}$ párosra legyen $f(k, r)$ az alábbi hátizsák feladat optimumértéke:

$$\frac{\sum_{j=1}^{k} a_j x_j \leq r \quad x_j \in \{0,1\},\ (j = 1, \ldots, k)}{\sum_{j=1}^{k} c_j x_j = z \to \max}$$

Akkor $k = m$ és $r = b$ esetében $f(m, b)$ pontosan a tekintett fela-

dat optimumát adja, így az optimum meghatározásához elegendő az $f(k,r)$ függvényt kiszámítani az adott értelmezési tartományra. Nyilvánvaló, hogy

$$f(k,0)=0, \quad (k=1,\ldots,m)$$

és

$$f(1,r)=\begin{cases} c_1, & \text{ha } a_1 \le r, \\ 0 & \text{különben.} \end{cases}, \ (r=0,1,\ldots,b)$$

Most tegyük fel, hogy ismertek az $f(k-1,r)$, $(r=0,1,\ldots,b)$ értékek valamely $k \ge 2$-re. Az $f(k,r)$-nek megfelelő feladatot oldjuk meg úgy, hogy $x_k=0$ és $x_k=1$ szerint szétbontjuk a lehetséges megoldások L' halmazát L'_0 és L'_1-re, majd a két maximum közül a nagyobbikat vesszük. Az $x_k=0$ helyettesítéssel L'_0-t az alábbi feladat határozza meg:

$$\begin{gathered} \textstyle\sum_{j=1}^{k-1} a_j x_j \le r \\ x_j \in \{0,1\}, \ (j=1,\ldots,k-1) \\ \hline \textstyle\sum_{j=1}^{k-1} c_j x_j = z \to \max \end{gathered}$$

Vegyük észre, hogy ez pontosan az $f(k-1,r)$-hez tartozó feladat, így az optimuma $f(k-1,r)$.

Az $x_k=1$ helyettesítéssel L'_1-et az alábbi feladat határozza meg:

$$\begin{gathered} \textstyle\sum_{j=1}^{k-1} a_j x_j \le r-a_k \\ x_j \text{ bináris } (j=1,\ldots,k-1) \\ \hline c_k+\textstyle\sum_{j=1}^{k-1} c_j x_j = z \to \max \end{gathered}$$

Ha $r<a_k$, akkor $L'_1=\emptyset$. Ha $r \ge a_k$, akkor a fenti feladat megegyezik az $f(k-1,r-a_k)$-hoz tartozó feladattal azzal az eltéréssel, hogy a célfüggvényben van egy c_k additív konstans. Tehát ebben az esetben a feladat optimuma $c_k+f(k-1,r-a_k)$. Terjesszük ki az $f(k,r)$ függvényt úgy, hogy legyen $f(k,r)=-W$, ha $r<0$. Akkor az eddigiek alapján

$$f(k,r)=\max\{f(k-1,r), c_k+f(k-1,r-a_k)\},$$

mely összefüggés alapján már kiszámíthatók az $f(k,r)$ értékek az f függvény értelmezési tartományán.

8.3. példa.

$$\begin{gathered} 2x_1+x_2+2x_3+x_4 \le 3 \\ x_j \in \{0,1\}, \ (j=1,2,3,4) \\ \hline 3x_1+2x_2+x_3+2x_4=z \to \max \end{gathered}$$

A számításokat hajtsuk végre egy $m \times (b+1)$-szeres táblázatban, ahol a k-adik sor és r-edik oszlop metszetében levő elem legyen $f(k,r)$. Akkor a táblázat első oszlopa 0-kat tartalmaz az $f(k,0) = 0, \quad (k = 1, \ldots, m)$ összefüggés alapján és az első sor elemei is egyszerűen meghatározhatók az

$$f(1,r) = \begin{cases} c_1, & \text{ha } a_1 \leq r, \\ 0 & \text{különben.} \end{cases}, \ (r = 0, 1, \ldots, b)$$

összefüggés szerint. Ezek után az

$$f(k,r) = \max\{f(k-1,r), c_k + f(k-1, r-a_k)\}$$

egyenlettel rendre ki tudjuk számítani a második oszlop, harmadik oszlop,..., $b+1$-edik oszlop elemeit. A tekintett példára az alábbi táblázat adódik.

	0	1	2	3
1	0	0	3	3
2	0	2	3	5
3	0	2	3	5
4	0	2	4	5

Most $f(4,3) = 5$, ezért az optimum értéke 5. Ennek ismeretében visszafelé haladva és az $f(k,r) = \max\{f(k-1,r), c_k + f(k-1, r-a_k)\}$ egyenletet használva, meghatározhatók az optimális megoldások is. A 8.3. példának két optimális megoldása van, az $(1,0,0,1)$ és az $(1,1,0,0)$ vektorok.

A téma befejezéseként a 7. fejezetre alapozva, felépítünk egy Branch-and-Bound eljárást a hátizsák probléma megoldására.

Branch-and-Bound eljárás a hátizsák feladat megoldására

Tekintsük a következő hátizsák feladatot

$$\frac{\sum_{j=1}^{m} a_j x_j \leq b \quad x_j \in \{0,1\}, \ (j = 1, \ldots, m)}{\sum_{j=1}^{m} c_j x_j = z \to \max}$$

amelyben minden együttható pozitív egész. Ha a tekintett feladatban minden $1 \leq j \leq m$ indexre kicseréljük az $x_j \in \{0,1\}$ feltételt a $0 \leq x_j \leq 1$ feltételre, akkor egy lineáris programozási feladatot kapunk, amelyet a tekintett *hátizsák feladat lineáris programozási relaxációjának* nevezünk. Az új feladat a következő:

$$\frac{\sum_{j=1}^{m} a_j x_j \leq b \quad 0 \leq x_j \leq 1, \ (j = 1, \ldots, m)}{\sum_{j=1}^{m} c_j x_j = z \to \max}$$

Jelölje L^* a tekintett hátizsák feladat lehetséges megoldásainak a halmazát és L a relaxáció lehetséges megoldásainak a halmazát. Akkor nyilvánvalóan $L^* \subseteq L$, amiből

$$\max\{z(\mathbf{x}) : \mathbf{x} \in L^*\} \leq \max\{z(\mathbf{x}) : \mathbf{x} \in L\}$$

következik. Ez pontosan azt jelenti, hogy a relaxáció optimuma felső korlátja a hátizsák feladat optimumának. Ennél több is igaz, a relaxáció optimumának egész része is felső korlátja a hátizsák feladat optimumának. Ezt az észrevételt fogjuk felhasználni a felépítésre kerülő Branch-and-Bound eljárás korlátozó függvényének meghatározásakor.

Ezt megelőzően megmutatjuk, hogy a relaxáció egy optimális megoldása nagyon egyszerűen meghatározható.

A változók indexezésének megváltoztatásával elérhető, hogy

$$c_1/a_1 \geq c_2/a_2 \geq \cdots \geq c_m/a_m$$

teljesüljön. Mivel minden együttható pozitív, ezért ezek a hányadosok léteznek. Ekkor érvényes a következő G. B. Dantzigtól [40] származó állítás.

8.2. segédtétel ([40]). *Jelölje k a legnagyobb olyan egész számot, amelyre $\sum_{t=1}^{k} a_k \leq b$ teljesül. Akkor az $\bar{x}_j = 1$, $(j = 0, \ldots, k)$, $\bar{x}_{k+1} = (b - \sum_{t=1}^{k} a_t)/a_{k+1}$ vektor egy optimális megoldása a tekintett hátizsák feladat lineáris programozási relaxációjának.*

Bizonyítás. Helyettesítsük a relaxációban az x_j változókat az y_j/a_j kifejezésekkel. Akkor egyszerűen belátható, hogy a helyettesítéssel előálló, alábbi feladat ekvivalens a tekintett relaxációval.

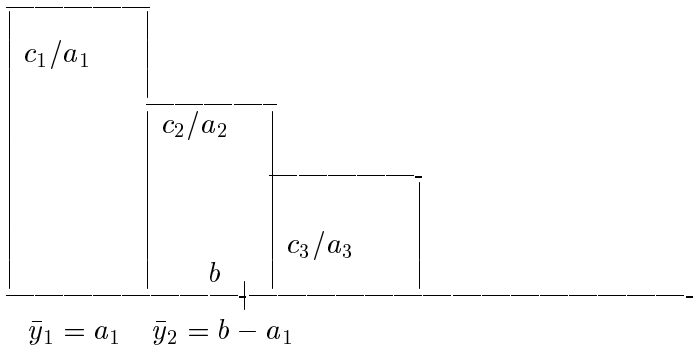
$$\begin{gathered} \textstyle\sum_{j=1}^{m} y_j \leq b \\ 0 \leq y_j \leq a_j,\ (j = 1, \ldots, m) \\ \hline \textstyle\sum_{j=1}^{m} (c_j/a_j) y_j = z \to \max \end{gathered}$$

Másrészt a kapott feladat optimális megoldása:

$$\bar{y}_j = a_j,\ (j = 1, \ldots, k),\ \bar{y}_{k+1} = b - \textstyle\sum_{t=1}^{k} a_t.$$

Ez utóbbi állítást a következő, területekre vonatkozó meggondolással láthatjuk be. Tekintsük a $\sum_{j=1}^{m} (c_j/a_j) y_j$ tagjait olyan téglalapoknak, amelyeknek az alapja y_j és a magassága c_j/a_j minden $j = 1, \ldots, m$ indexre. Ekkor a célfüggvény értéke ezen területek összege. Most vegyünk fel egy horizontális tengelyt és vegyük fel rá rendre az $a_1, \ldots, a_m$ hosszúságú intervallumokat. Nyilván akkor kapjuk a legnagyobb összeget, ha a magas

téglalapok alapja a lehető legnagyobb. Ezt úgy tudjuk elérni, hogy felvisszük b-t erre a tengelyre és $y_1, y_2, \ldots$ felveszik a lehető legnagyobb értéküket egészen addig, amíg összegük el nem éri b-t. A 8.1. ábra egy olyan szituációt mutat be, amikor y_1 feveszi a lehető legnagyobb értékét, ami a_1, és y_2 a $b - a_1$ értéket kapja, mivel $a_1 + a_2 > b$.



8.1. ábra. A maximális terület meghatározása.

Az új feladat $\bar{y}_j = a_j$, $(j = 1, \ldots, k)$, $\bar{y}_{k+1} = b - \sum_{t=1}^{k} a_t$ optimális megoldásából az $\bar{y}_j = a_j \bar{x}_j$ helyettesítéssel megkapjuk a kiindulási feladat optimális megoldását. A helyettesítés eredménye:

$$\bar{x}_j = 1,\ (j = 1 \ldots, k),\ \bar{x}_{k+1} = (b - \textstyle\sum_{t=1}^{k} a_t)/a_{k+1},$$

ami pontosan a 8.2. segédtételben megadott vektor. Ezzel a 8.2. segédtétel igazolását befejeztük.

A lineáris programozási relaxáció optimális megoldása meghatározásának bemutatására tekintsük a következő példát.

8.4. példa.

$$x_1 + 4x_2 + 3x_3 + 2x_4 + 6x_5 + x_6 \leq 8$$

$$0 \leq x_j \leq 1,\ (j = 1, \ldots, 6)$$

$$3x_1 + x_2 + 4x_3 + 2x_4 + x_5 + 5x_6 = z \to \max$$

$$\frac{c_6}{a_6} \geq \frac{c_1}{a_1} \geq \frac{c_3}{a_3} \geq \frac{c_4}{a_4} \geq \frac{c_2}{a_2} \geq \frac{c_5}{a_5}$$

$$5 \ \geq \ 3 \ \geq \ \frac{4}{3} \ \geq \ 1 \ \geq \ \frac{1}{4} \ \geq \ \frac{1}{6}$$

A relaxáció optimális megoldása:

$$\bar{x}_6 = 1,\ \bar{x}_1 = 1,\ \bar{x}_3 = 1,\ \bar{x}_4 = 1,\ \bar{x}_2 = 1/4.$$

8.1. megjegyzés. Vegyük észre, hogy amennyiben a hátizsák feladat lineáris programozási relaxációjának optimális megoldása egész, akkor ez a bináris vektor optimális megoldása a hátizsák feladatnak is. Ha a relaxációnak a 8.2. segédtétel alapján meghatározott optimális megoldása nem egész, akkor ezen optimális megoldásban az egyetlen nem egész komponenst 0-ra változtatva, a hátizsák feladat egy jó lehetséges megoldásához jutunk

Például a 8.4. példa esetében a hátizsák feladat egy jó lehetséges megoldása:

$x_6^* = 1,\ x_1^* = 1,\ x_3^* = 1,\ x_4^* = 1,\ x_j^* = 0$, a többi indexekre.

B&B eljárás

Az eljárást a 7. fejezetben történt tárgyalásnak megfelelően építjük fel.

I. Az Ω halmaz meghatározása

A (8.1) feladat esetében az Ω halmazt az összes lehetséges bináris m-esek halmazaként definiáljuk, azaz

$$\Omega = \{(x_1, \ldots, x_m) : x_j \in \{0, 1\}, j = 1, \ldots, m\}.$$

Nyilvánvalóan $L \subseteq \Omega$. Másrészt $|\Omega| = 2^m$, így Ω véges.

II. A φ szétválasztási függvény és a g korlátozó függvény megadása

1. Korlátozó függvény

A g korlátozó függvényt Ω speciális részhalmazaira fogjuk definiálni. Ehhez legyen $I \subseteq \{1, \ldots, m\}$, $J \subseteq \{1, \ldots, m\}$, $I \bigcap J = \emptyset$. Akkor

$$\Omega_{I,J} = \{\mathbf{x} : \mathbf{x} \in \Omega \ \&\ x_i = 1, \ \text{ha}\ i \in I \ \&\ x_j = 0, \ \text{ha}\ j \in J\}.$$

Vegyük észre, hogy $\Omega_{I,J}$ definiálásakor a változók egy részét konstans értéken rögzítettük. A rögzített értékű változókat behelyettesítve a (8.1) feladatba, ismételten egy hátizsák feladathoz jutunk. Nevezzük ezt a feladatot (8.1) (I, J)-részproblémájának, és jelölje ezen részprobléma lehetséges megoldásainak halmazát $L_{I,J}^*$, ahol a lehetséges megoldásokba a rögzített értékű változókat is beleértjük. Egyszerűen belátható, hogy

$$L^*_{I,J} = L^* \bigcap \Omega_{I,J}.$$

Most legyen az (I, J)-részprobléma lineáris programozási relaxációjának optimuma $z_{I,J}$. Akkor nyilvánvaló, hogy $z_{I,J}$ egész része felső korlátja a $z(\mathbf{x})$ $(\mathbf{x} \in L^* \bigcap \Omega_{I,J})$ célfüggvényértékeknek, így ezt rendelve az $\Omega_{I,J}$ halmazhoz, a korlátozó függvényünk rendelkezik a kívánt tulajdonságokkal. Előfordulhat, hogy a rögzített értékű változókkal olyan hátizsák feladathoz jutunk, amelynek jobboldala negatív. Ekkor az illető részproblémának nincsen lehetséges megoldása, így a $-W$ konstanst kell az $\Omega_{I,J}$ halmazhoz rendelni.

A 8.1. megjegyzés alapján tudjuk, hogy amennyiben az (I, J)-részprobléma lineáris programozási relaxációjának optimális megoldása egész, úgy a részprobléma optimális megoldásához jutunk. Ellenkező esetben pedig tudunk képezni egy lehetséges megoldást.

2. Szétválasztási függvény

Legyen $\Omega_{I,J}$ a fentiekben definiált tetszőleges halmaz, ahol $I, J \subseteq \{1, \ldots, m\}$, és $I \bigcap J = \emptyset$. Nyilvánvaló, hogy $|\Omega_{I,J}| > 1$ akkor és csak akkor teljesül, ha $|I| + |J| < m$. Tegyük fel, hogy az utóbbi reláció fennáll. Akkor van olyan $k \in \{1, \ldots, m\}$, hogy $k \notin I \bigcup J$. Legyen $I_1 = I \bigcup \{k\}$, $J_1 = J$, $I_2 = I$, $J_2 = J \bigcup \{k\}$. Egyszerűen belátható, hogy $\{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}$ az $\Omega_{I,J}$ halmaz egy valódi osztályozása. Most legyen

$$\varphi(\Omega_{I,J}) = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}.$$

A fentiekben formalizált szétválasztás során egy további változónak, x_k-nak az értékét rögzítjük 0 és 1 értéken, és ennek megfelelően bontjuk fel két osztályra a tekintett $\Omega_{I,J}$ halmazt.

A szétválasztási függvény definiálásához egy további pontosítás szükséges. Nevezetesen, meg kell adnunk k kiválasztási stratégiáját. Ezt rögzítsük oly módon, hogy legyen k azon komponens indexe, amely az illető részprobléma relaxációja optimális megoldásában nem egész értéket kap.

Faépítési stratégia

Alkalmazzuk a maximális korláttal rendelkező szögpont kiválasztásának stratégiáját.

Ahhoz, hogy teljessé tegyük a $B\&B$ eljárást, meg kell adnunk egy heurisztikát egy induló megoldás meghatározásához. Az egyszerűség kedvéért válasszuk azt a heurisztikát, amely a kiindulási feladat relaxációjából képezhető lehetséges megoldást szolgáltatja.

Mivel a 7. fejezetben minimum feladatok megoldására építettünk fel egy eljárást, most pedig maximum feladatokat akarunk megoldani, ezért részletesen megadjuk az eljárást.

Eljárás

Előkészítő rész. Oldjuk meg a tekintett hátizsák feladat lineáris programozási relaxációját, majd a kapott optimális megoldásból határozzuk meg a hátizsák feladat egy $\mathbf{x}_0$ lehetséges megoldását. Legyen a $\bar{z}$ változó értéke $z(\mathbf{x}_0)$ és az $\mathbf{x}^*$ vektorváltozó értéke $\mathbf{x}_0$. Legyen $g(\Omega)$ a relaxáció optimumának egész része. Legyen $r = 0$ és

$$F_0 = \begin{cases} \{\Omega\}, & \text{ha } g(\Omega) > \bar{z}, \\ \emptyset & \text{különben.} \end{cases}$$

Ezt követően folytassuk az eljárást az iterációs eljárásrésszel.

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $F_r = \emptyset$, akkor vége az eljárásnak; $\mathbf{x}^*$ a feladat optimális megoldása, $\bar{z}$ az optimum értéke. Különben folytassuk a 2. lépéssel az eljárást.

- *2. lépés.* Ha $F_r \neq \emptyset$, akkor válasszunk ki egy, a legnagyobb korláttal rendelkező elemet F_r elemei közül, jelölje ezt $\Omega_{I,J}$. Alkalmazzuk a φ szétválasztási függvényt $\Omega_{I,J}$-re. Legyen

  $$\varphi(\Omega_{I,J}) = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}.$$

  Határozzuk meg az Ω_{I_1,J_1}, Ω_{I_2,J_2} halmazokra a $g(\Omega_{I_1,J_1})$, $g(\Omega_{I_2,J_2})$ korlátokat. Jelölje $\mathbf{x}_1, \mathbf{x}_2$ a korlátok meghatározása során előálló lehetséges megoldásokat, feltéve, hogy vannak ilyenek. Legyen z_r a $z(\mathbf{x}_1)$ és $z(\mathbf{x}_2)$ függvényértékek maximuma, és $\mathbf{x}^{(r)}$ az $\mathbf{x}_1$ és $\mathbf{x}_2$ vektorok közül az, amelyen a z függvény z_r értéket vesz fel. Definiáljuk újra $\bar{z}$-t és $\mathbf{x}^*$-ot a következők szerint. Ha $z_r \leq \bar{z}$, akkor $\bar{z}$ és $\mathbf{x}^*$ nem változik, míg $z_r > \bar{z}$ esetén $\bar{z}$-nak adjuk a z_r értéket és $\mathbf{x}^*$-ot változtassuk $\mathbf{x}^{(r)}$-re. Ezek után legyen

$$F_{r+1} = \{\Omega_{I',J'} : \Omega_{I',J'} \in (F_r \setminus \{\Omega_{I,J}\}) \cup \{\Omega_{I_1,J_1} \cup \Omega_{I_2,J_2}\} \ \& \ g(\Omega_{I',J'}) > \bar{z}\}.$$

Növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

Az eljárás végrehajtásának demonstrálására tekintsük a következő példát.

8.5. példa.

$$\begin{gathered} 4x_1 + 3x_2 + 7x_3 + 7x_4 + 12x_5 \leq 20 \\ x_j \in \{0,1\},\ (j = 1,2,3,4,5) \\ \hline 3x_1 + 2x_2 + 4x_3 + 3x_4 + 5x_5 = z \to \max \end{gathered}$$

Eljárás.

Előkészítő rész.

A relaxáció optimális megoldása: $(1,1,1,6/7,0)$. Ekkor az ebből képezhető lehetséges megoldás $\mathbf{x}_0 = (1,1,1,0,0)$, így $\bar{z} = z(\mathbf{x}_0) = 9$, $\mathbf{x}^* = \mathbf{x}_0$, $g(\Omega_{\emptyset,\emptyset}) = 11$, $r = 0$, $F_0 = \{\Omega_{\emptyset,\emptyset}\}$.

Iterációs rész. (0-adik iteráció.)

1. lépés. Mivel $F_0 = \{\Omega_{\emptyset,\emptyset}\}$, ezért a 2. lépés következik.

2. lépés. F_0-ból csak $\Omega_{\emptyset,\emptyset}$ választható. Mivel a relaxáció optimális megoldásában az x_4 változó kapott nem egész értéket, ezért a szétválasztásnál x_4-nek kell értékeket adni, azaz

$$\varphi(\Omega_{\emptyset,\emptyset}) = \{\Omega_{\{4\},\emptyset}, \Omega_{\emptyset,\{4\}}\}.$$

A $(\{4\},\emptyset)$-részfeladat relaxációjának optimális megoldása az $(1,1,6/7,1,0)$ vektor. Az optimum egész része 11, ezért $g(\Omega_{\{4\},\emptyset}) = 11$, a származtatható lehetséges megoldás $\mathbf{x}_1 = (1,1,0,1,0)$, $z(\mathbf{x}_1) = 8$.

Az $(\emptyset,\{4\})$-részfeladat relaxációjának optimális megoldása az $(1,1,1,0,1/2)$ vektor. Az optimum egész része 11, és ezért $g(\Omega_{\emptyset,\{4\}}) = 11$. $\mathbf{x}_2 = (1,1,1,0,0)$, $z(\mathbf{x}_2) = 9$. Mivel $z_0 = 9$, ezért $\bar{z}$ és $\mathbf{x}^*$ nem változnak. $F_1 = \{\Omega_{\{4,\},\emptyset}, \Omega_{\emptyset,\{4\}}\}$, $r = r + 1$.

Iterációs rész. (1. iteráció.)

1. lépés. Mivel $F_1 \neq \emptyset$, ezért a 2. lépés következik.

2. lépés. Válasszuk az $\Omega_{\{4\},\emptyset}$ halmazt F_2-ből. Ezt az x_3 változó értékadásaival kell szétválasztani.

$$\varphi(\Omega_{\{4\},\emptyset}) = \{\Omega_{\{3,4\},\emptyset}, \Omega_{\{4\},\{3\}}\}.$$

A $(\{3,4\},\emptyset)$-részfeladat relaxációjának optimális megoldása az $(1, 2/3, 1, 1, 0)$ vektor. Az optimum egész része 11, ezért $g(\Omega_{\{3,4\},\emptyset}) = 11$, a származtatható lehetséges megoldás a $\mathbf{x}_1 = (1, 0, 1, 1, 0)$ vektor és $z(\mathbf{x}_1) = 10$.

A $(\{4,\},\{3\})$-részfeladat relaxációjának optimális megoldása az $(1, 1, 0, 1, 1/2)$ vektor. Az optimum egész része 10, ezért $g(\Omega_{\{4\},\{3\}}) = 10$, a származtatható lehetséges megoldás az $\mathbf{x}_2 = (1, 1, 0, 1, 0)$ vektor és $z(\mathbf{x}_2) = 8$. Most $z_1 > \bar{z}$, ezért $\bar{z}$-t z_1-re, azaz 10-re, és $\mathbf{x}^*$-ot $\mathbf{x}_1$-re változtatjuk, $F_2 = \{\Omega_{\{3,4\},\emptyset}, \Omega_{\emptyset,\{4\}}\}$, $r = r + 1$.

Iterációs rész. (2. iteráció.)

1. lépés. $F_2 \neq \emptyset$, ezért a 2. lépés következik.
2. lépés. Válasszuk F_2-ből az $\Omega_{\{3,4\},\emptyset}$ halmazt. Ezt az x_2 változónak adott értékadással kell szétválasztani.

$$\varphi(\Omega_{\{3,4\},\emptyset}) = \{\Omega_{\{2,3,4\},\emptyset}, \Omega_{\{3,4\},\{2\}}\}.$$

A $(\{2,3,4\},\emptyset)$-részfeladat relaxációjának optimális megoldása: $(3/4, 1, 1, 1, 0)$, az optimum értéke 11, így $g(\Omega_{\{2,3,4\},\emptyset}) = 11$. Származtatható megoldás: $\mathbf{x}_1 = (0, 1, 1, 1, 0)$, $z(\mathbf{x}_1) = 9$.

A $(\{3,4\},\{2\})$-részfeladat relaxációjának optimális megoldása: $(1, 0, 1, 1, 1/6)$, az optimum értéke 10, így $g(\Omega_{\{3,4\},\{2\}}) = 10$. Származtatható megoldás: $\mathbf{x}_2 = (1, 0, 1, 1, 0)$, $z(\mathbf{x}_2) = 10$. $z_2 \leq \bar{z}$, ezért $\bar{z}$ és $\mathbf{x}^*$ nem változnak. $F_3 = \{\Omega_{\{2,3,4\},\emptyset}, \Omega_{\emptyset,\{4\}}\}$, $r = r + 1$.

Iterációs rész. (3. iteráció.)

1. lépés. $F_3 \neq \emptyset$, ezért a 2. lépés következik.
2. lépés. Válasszuk F_3-ból az $\Omega_{\emptyset,\{4\}}$ halmazt. Ezt az x_5 változónak adott értékadással kell szétválasztani.

$$\varphi(\Omega_{\emptyset,\{4\}}) = \{\Omega_{\{5\},\{4\}}, \Omega_{\emptyset,\{4,5\}}\}.$$

Az $(\{5\},\{4\})$-részfeladat relaxációjának optimális megoldása: $(1, 1, 1/7, 0, 1)$, az optimum értéke 10, így $g(\Omega_{\{5\},\{4\}}) = 10$. Származtatható megoldás: $\mathbf{x}_1 = (1, 1, 0, 0, 1)$, $z(\mathbf{x}_1) = 10$.

Az $(\emptyset, \{4,5\})$-részfeladat relaxációjának optimális megoldása: $(1,1,1,0,0)$, az optimum értéke 9, így $g(\Omega_{\emptyset,\{4,5\}}) = 9$. Származtatható megoldás: $\mathbf{x}_2(1,1,1,0,0)$, $z(\mathbf{x}_2) = 9$. $z_3 \leq \bar{z}$, ezért $\bar{z}$ és $\mathbf{x}^*$ nem változnak. $F_4 = \{\Omega_{\{2,3,4\},\emptyset}\}$, $r = r + 1$.

Iterációs rész. (4. iteráció.)

1. lépés. $F_4 \neq \emptyset$, ezért a 2. lépés következik.
2. lépés. Válasszuk F_4-ból az $\Omega_{\{2,3,4\},\emptyset}$ halmazt. Ezt az x_1 változónak adott értékadással kell szétválasztani.

$$\varphi(\Omega_{\{2,3,4\},\emptyset}) = \{\Omega_{\{1,2,3,4\},\emptyset}, \Omega_{\{2,3,4\},\{1\}}\}.$$

Az $(\{1,2,3,4\}, \emptyset)$-részfeladat jobboldala negatív, ezért $g(\Omega_{\{1,2,3,4\},\emptyset}) = -W$.
A $(\{2,3,4\}, \{1\})$-részfeladat relaxációjának optimális megoldása: $(0,1,1,1,1/4)$, az optimum értéke 10, így $g(\Omega_{\{2,3,4\},\{1\}}) = 10$. Származtatható megoldás: $\mathbf{x}_2(0,1,1,1,0)$, $z(\mathbf{x}_2) = 9$. $z_4 \leq \bar{z}$, ezért $\bar{z}$ és $\mathbf{x}^*$ nem változnak. $F_5 = \emptyset$, $r = r + 1$.

Iterációs rész. (5. iteráció.)

1. lépés. Mivel $F_5 = \emptyset$, ezért vége az eljárásnak. Az optimális megoldás: $\mathbf{x}^* = (1,0,1,1,0)$, az optimum: 10.

A hátizsák feladat megoldására számos Branch-and-Bound eljárást építettek fel. Az ilyen eljárások iránt érdeklődők számára a teljesség igénye nélkül ajánljuk a következő munkákat: [11], [55], [81], [90], [92], [110], [133], [134], [146].

9. Korlátos egészértékű lineáris programozási feladat

Tekintsük az alábbi egészértékű lineáris programozási feladatot:

$$\begin{array}{l} \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \text{ és } \mathbf{x} \text{ egész} \\ \hline \alpha + \mathbf{c}\mathbf{x} = z \to \min \end{array} \tag{9.1}$$

A feladatot az alábbi feltételek mellett fogjuk vizsgálni:

(i) az $\mathbf{A}$ mátrix elemei, a $\mathbf{c}$ vektor komponensei, valamint α rendre egészek,

(ii) a $\mathbf{b}$ vektor komponensei nemnegatív egészek,

(iii) a lehetséges megoldások L halmaza korlátos, és a korlátosság a feltételrendszerben szereplő $\sum_{t=1}^{m} x_t \leq k$ feltétellel van biztosítva, ahol k pozitív egész.

Nyilvánvaló, hogy az adott feltételek mellett L véges, nemüres halmaz és a (9.1) feladatnak létezik optimális megoldása.

A fenti modellel számos gyakorlati probléma leírható, amelyek közül az alábbiakban ismertetünk néhányat.

Beruházási probléma

Adott egy üzem, amely különböző beruházásokkal tudja bővíteni a termelését. Az egyes beruházások megvalósítása több évig tart. Az évente rendelkezésre álló beruházási tőke csak korlátozott számú beruházást tesz lehetővé. A lehetőségek közül válasszuk ki a beruházások azon csoportját, amely a rendelkezésre álló beruházási forrásból megvalósítható, és maximális nyereséget eredményez.

Jelölje

m a lehetséges beruházások számát,
n a beruházási terv éveinek a számát,
a_{ij} a j-edik beruházás tőkeigényét az i-edik évben ($1 \leq i \leq n$, $1 \leq j \leq m$),
b_i az i-edik évben beruházásra felhasználható tőkét ($i = 1, \ldots, n$),

c_j a j-edik beruházás megvalósításával keletkező nyereséget $(j = 1, \ldots, m)$.

Legyen tetszőleges $1 \leq j \leq m$ egészre

$$x_j = \begin{cases} 1, & \text{ha a } j\text{-edik beruházás megvalósításra kerül,} \\ 0 & \text{különben.} \end{cases}$$

Akkor az i-edik évben $\sum_{j=1}^{m} a_{ij}x_j$ tőke kerül felhasználásra, amely nem haladhatja meg a rendelkezésre álló b_i mennyiséget. Így a megvalósíthatósághoz a

$$\sum_{j=1}^{m} a_{ij}x_j \leq b_i \quad (i = 1, \ldots, n),$$

$$x_j \in \{0, 1\} \quad (j = 1, \ldots, m)$$

feltételeknek kell teljesülnie. A beruházásokkal keletkező teljes nyereség: $\sum_{j=1}^{m} c_j x_j$, amelynek a maximumát keressük a fenti feltételek mellett.

Hátizsák probléma

Ezzel a problémával az előző fejezetben részletesen foglalkoztunk.

Hajórakodási probléma

Egy hajó rakterére súly- és térfogatkorlátozás van előírva. A hajót különböző típusú bálákkal kívánjuk megrakni, az egyes típusokra azonos a súly és a térfogat. Határozzuk meg a maximális értékű rakományt.

Jelölje

m a bálatípusok számát,
a_{1j} a j-edik bálatípus egy bálájának súlyát $(j = 1, \ldots, m)$,
a_{2j} a j-edik bálatípus egy bálájának térfogatát $(j = 1, \ldots, m)$,
c_j a j-edik bálatípus egy bálájának értékét $(j = 1, \ldots, m)$,
b_1 a hajó súlykorlátját,
b_2 a hajó térfogatkorlátozását,
x_j a j-edik bálatípusból behajózásra kerülő darabszámot $(j = 1, \ldots, m)$.

Akkor a rakomány súlya $\sum_{j=1}^{m} a_{1j}x_j$, a rakomány térfogata $\sum_{j=1}^{m} a_{2j}x_j$, amelyek rendre nem haladhatják meg a b_1 és b_2 korlátokat. Így a következő feltételeknek kell teljesülnie:

$$\sum_{j=1}^{m} a_{ij}x_j \leq b_i \quad (i = 1, 2),$$

$$x_j \geq 0 \text{ és } x_j \text{ egész} \quad (j = 1, \ldots, m).$$

A rakomány értéke: $\sum_{j=1}^{m} c_j x_j$, amelynek a maximumát keressük a fenti feltételek mellett.

Vegyük észre, hogy a tekintett problémák mindegyike (9.1) típusú feladathoz vezet, ha az együtthatók egészek. Valóban, az első két feladatnál az $x_j \in \{0, 1\}$ $(j = 1, \ldots, m)$ feltételekből következik, hogy $\sum_{j=1}^{m} x_j \leq m$, amivel teljesül az előírt korlátosság. A harmadik problémánál feltéve, hogy $a_{ij} > 0$ $(i = 1, 2; j = 1, \ldots, m)$,

$$x_j \leq M_j = \min\{[\frac{b_1}{a_{1j}}], [\frac{b_2}{a_{2j}}]\},$$

így $\sum_{j=1}^{m} x_j \leq \sum_{j=1}^{m} M_j$, azaz ismét teljesül a korlátosság.

Visszatérve a (9.1) feladat vizsgálatához, az ilyen típusú feladatok megoldására elsőként R. E. Gomory [78] dolgozott ki egy egzakt eljárást 1958-ban, amely a *metsző síkok módszere* néven ismeretes. Az eljárással kapcsolatban R. G. Jeroslow és K. O. Kortanek [98] igazolták, hogy rögzített feladatméret mellett az eljárás iterációs lépésszáma bármilyen nagy lehet. Ezt követően D. S. Rubin [156] megadott egy olyan feladatosztályt, amely kétváltozós és egyetlen feltételből álló feladatokat tartalmaz, és tetszőleges n természetes számhoz létezik az osztálynak olyan feladata, hogy az illető feladatra alkalmazva a metsző síkok módszerét, az eljárás iterációs lépésszáma nem kisebb, mint n. A későbbiek során Gomory alapötletét alkalmazva, számos úgynevezett metszési eljárás került kidolgozásra, amelyekről rendre kiderült, hogy bizonyos feladatokat igen nagy iterációs lépésszámmal lehet ezen eljárások alkalmazásával megoldani.

A (9.1) feladat megoldására teljesen más technikát, egy Branch-and-Bound eljárást javasolt A. H. Land és A. G. Doig 1960-ban a már említett [119] dolgozatban. Az általuk leírt módszert módosította 1965-ben R. Dakin [39]. A következőkben ezt a módosított eljárást fogjuk felépíteni a (9.1) feladat megoldására. Mielőtt ezt megtennénk, szükségesek bizonyos előkészületek.

Vegyük a (9.1) feladatot, és elhagyva a feltételrendszerből a változók egészértékűségére vonatkozó kikötést, konstruáljuk meg a következő lineáris programozási feladatot.

$$\begin{array}{l} \mathbf{Ax} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \\ \hline \alpha + \mathbf{cx} = z \rightarrow \min \end{array} \tag{9.2}$$

A (9.2) feladatot szokásos a (9.1) *diszkrét feladathoz tartozó folytonos feladatnak* vagy *a diszkrét feladat lineáris programozási relaxációjának* nevezni. Jelölje a (9.2) feladat lehetséges megoldásainak a halmazát S. Akkor a feladatpár kapcsolatát illetően nyilvánvalóak a következők:

(i) a folytonos feladatnak az adott feltételek mellett létezik optimális megoldása,

(ii) $L \subseteq S$ és L pontosan az S halmaz egész koordinátájú pontjainak a halmaza,

(iii) $\min\{z(\mathbf{x}) : \mathbf{x} \in S\} \leq \min\{z(\mathbf{x}) : \mathbf{x} \in L\}$,

(iv) ha a folytonos feladat optimális megoldása egész, akkor ez optimális megoldása a diszkrét feladatnak is.

Részben a fentiek, részben a továbbiak demonstrálására tekintsük az alábbi konkrét feladatot.

9.1. példa

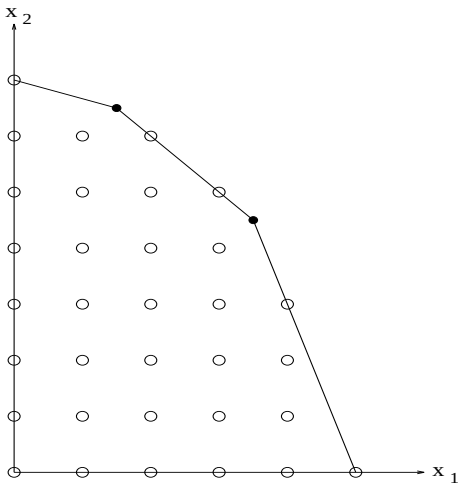
$$\begin{array}{r} x_1 + 3x_2 \leq 21 \\ x_1 + x_2 \leq 8 \\ 3x_1 + x_2 \leq 15 \\ x_1, x_2 \geq 0 \text{ , } x_1 \text{ és } x_2 \text{ egész} \\ \hline -2x_1 - 3x_2 = z \rightarrow \min \end{array}$$

A tekintett feladathoz tartozó folytonos feladat a következő:

$$\begin{array}{r} x_1 + 3x_2 \leq 21 \\ x_1 + x_2 \leq 8 \\ 3x_1 + x_2 \leq 15 \\ x_1, x_2 \geq 0 \\ \hline -2x_1 - 3x_2 = z \rightarrow \min \end{array}$$

A 9.1. ábra mutatja az S és L halmazokat. S a sokszög pontjainak a halmaza, míg L a sokszög egész koordinátájú pontjainak a halmaza. Az utóbbi pontokat karikákkal jelöltük meg.

A (9.1) feladat megoldására szolgáló eljárást a 7. fejezetben leírtak szerint fogjuk felépíteni.



9.1. ábra. Az S és L halmazok ábrázolása.

I. Az Ω halmaz meghatározása

A tekintett probléma esetében az Ω halmazt a következőképpen definiáljuk.

$$\Omega = \{(x_1, \ldots, x_m) : 0 \le x_j \le k,\ x_j \text{ egész},\ j = 1, \ldots, m\}\ .$$

Vegyük észre, hogy a (9.1) feladat tetszőleges $\bar{\mathbf{x}}$ lehetséges megoldására $\bar{x}_j \ge 0$, $\bar{x}_j$ egész $(j = 1, \ldots m)$, valamint $\sum_{j=1}^{m} \bar{x}_j \le k$ teljesül. De akkor $0 \le \bar{x}_j \le k$, $\bar{x}_j$ egész $(j = 1, \ldots, m)$, így $\bar{\mathbf{x}} \in \Omega$. Következésképp, $L \subseteq \Omega$. Másrészt Ω véges, ugyanis $|\Omega| = (k+1)^m$.

A vizsgált példában Ω a $(0,0)$, $(8,0)$, $(8,8)$, $(0,8)$ pontok által meghatározott négyzet egész koordinátájú pontjainak a halmaza, és $|\Omega| = 9^2$.

II. A φ szétválasztási függvény és a g korlátozó függvény megadása

1. Korlátozó függvény

A g függvényt az Ω halmaz speciális részhalmazaira fogjuk definiálni. Ehhez jelöljenek $\mathbf{e}$ és $\mathbf{f}$ olyan nemnegatív vektorokat, amelyeknek minden koordinátája egész, és $\mathbf{0} \le \mathbf{e} \le \mathbf{f} \le (k, \ldots, k)$ teljesül. Legyen

$$\Omega_{\mathbf{e},\mathbf{f}} = \{\mathbf{x} : \mathbf{x} \text{ egész és } \mathbf{e} \le \mathbf{x} \le \mathbf{f}\}\ .$$

Ekkor $\Omega_{\mathbf{e},\mathbf{f}} \subseteq \Omega$ és $\Omega_{\mathbf{e},\mathbf{f}}$ akkor és csak akkor egyelemű, ha $\mathbf{e} = \mathbf{f}$. Speciálisan, $\mathbf{e} = \mathbf{0}$ és $\mathbf{f} = (k, \ldots, k)$ esetén $\Omega_{\mathbf{e},\mathbf{f}} = \Omega$.

Legyen most $\Omega_{\mathbf{e},\mathbf{f}}$ az Ω halmaz tetszőleges, a fentieket kielégítő részhalmaza. Akkor $\bar{\mathbf{x}} \in L \cap \Omega_{\mathbf{e},\mathbf{f}}$ akkor és csak akkor teljesül, ha $\bar{\mathbf{x}}$ lehetséges megoldása az alábbi lineáris programozási feladatnak:

$$\begin{array}{l} \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{e} \leq \mathbf{x} \leq \mathbf{f} \\ \mathbf{x} \text{ egész} \\ \hline \alpha + \mathbf{c}\mathbf{x} = z \to \min \end{array} \tag{9.3}$$

Következésképp, az $L \cap \Omega_{\mathbf{e},\mathbf{f}}$ halmazt megadhatjuk, mint a (9.3) feladat lehetséges megoldásainak a halmazát.

Tekintsük most a (9.3) egészértékű feladathoz tartozó folytonos lineáris programozási feladatot. Ez a következő:

$$\begin{array}{l} \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{e} \leq \mathbf{x} \leq \mathbf{f} \\ \hline \alpha + \mathbf{c}\mathbf{x} = z \to \min \end{array} \tag{9.4}$$

A két feladat kapcsolatát illetően, jelölje (9.3) lehetséges megoldásainak halmazát L', és (9.4) lehetséges megoldásainak a halmazát S'. Akkor nyilvánvaló, hogy teljesülnek a következők:

(1) $L' \subseteq S'$ és L' pontosan S' egész koordinátájú pontjainak a halmaza,

(2) ha $L' \neq \emptyset$, akkor $\min\{z(\mathbf{x}) : \mathbf{x} \in S'\} \leq \min\{z(\mathbf{x}) : \mathbf{x} \in L'\}$.

Most a (2) tulajdonság alapján definiálhatjuk a g korlátozó függvényt a következők szerint:

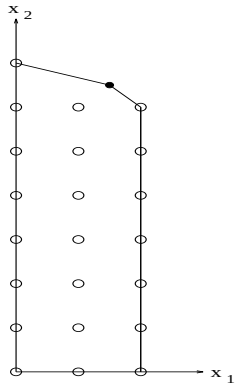
$$g(\Omega_{\mathbf{e},\mathbf{f}}) = \begin{cases} \min\{z(\mathbf{x}) : \mathbf{x} \in S'\}, & \text{ha } S' \neq \emptyset \text{ és } \mathbf{e} \neq \mathbf{f}, \\ z(\mathbf{e}), & \text{ha } S' \neq \emptyset \text{ és } \mathbf{e} = \mathbf{f}, \\ W & \text{különben .} \end{cases}$$

Egyszerű esetvizsgálattal igazolható, hogy g kielégíti a 7. fejezetben előírt feltételeket. Valóban, $|\Omega_{\mathbf{e},\mathbf{f}}| > 1$ esetén $\mathbf{e} \neq \mathbf{f}$, de akkor tetszőleges $\bar{\mathbf{x}} \in L \cap \Omega_{\mathbf{e},\mathbf{f}}$-re $g(\Omega_{\mathbf{e},\mathbf{f}}) \leq z(\bar{\mathbf{x}})$. Ha $|\Omega_{\mathbf{e},\mathbf{f}}| = 1$, akkor $\mathbf{e} = \mathbf{f}$, és $L \cap \Omega_{\mathbf{e},\mathbf{f}} \neq \emptyset$ esetén $S' = \{\mathbf{e}\} = L \cap \Omega_{\mathbf{e},\mathbf{f}}$ és $g(\Omega_{\mathbf{e},\mathbf{f}}) = z(\mathbf{e})$.

Tekintsük ismét az előző példát. Legyen $\mathbf{e} = (0,0)$, $\mathbf{f} = (2,8)$. Akkor $\Omega_{\mathbf{e},\mathbf{f}}$ a $(0,0)$, $(2,0)$, $(2,8)$, $(0,8)$ pontok által meghatározott téglalap egészértékű pontjainak a halmaza. Az $L \cap \Omega_{\mathbf{e},\mathbf{f}}$ halmazt leíró lineáris programozási feladat a következő:

$$
\begin{array}{r}
x_1 \ +3x_2 \leq 21 \\
x_1 \ +x_2 \ \leq 8 \\
3x_1 \ +x_2 \leq 15 \\
0 \leq x_1 \leq 2,\ x_1 \text{ egész} \\
0 \leq x_2 \leq 8,\ x_2 \text{ egész} \\
\hline
-2x_1 - 3x_2 = z \rightarrow \min
\end{array}
$$

A 9.2. ábra mutatja az L' és S' halmazokat. S' a sokszög pontjainak a halmaza, L' a sokszög egész koordinátájú pontjainak a halmaza.



9.2. ábra. Az L' és S' halmazok ábrázolása.

A folytonos feladat optimális megoldása : $(3/2, 13/2)$, az optimum értéke: $-45/2$. Így

$$g(\Omega_{\mathbf{e},\mathbf{f}}) = -\frac{45}{2}\,.$$

2. Szétválasztási függvény

Legyen $\Omega_{\mathbf{e},\mathbf{f}}$ tetszőleges, ahol $\mathbf{0} \leq \mathbf{e} \leq \mathbf{f} \leq (k, \ldots, k)$, $\mathbf{e} \neq \mathbf{f}$. (A 0-adik iterációs lépésben $\mathbf{e} = \mathbf{0}$ és $\mathbf{f} = (k, \ldots, k)$). Most tegyük fel, hogy a megfelelő (9.4) folytonos lineáris programozási feladat $\bar{\mathbf{x}}$ optimális megoldása nem egész. Csak erre az esetre fogjuk a φ függvényt definiálni. Ekkor $\bar{\mathbf{x}}$ -nak valamelyik komponense nem egész. Jelölje az első ilyen komponenst $\bar{x}_i$. Ez esetben $e_i < \bar{x}_i < f_i$. Képezzük a következő $\mathbf{e}^{(1)}$, $\mathbf{f}^{(1)}$, $\mathbf{e}^{(2)}$, $\mathbf{f}^{(2)}$ vektorokat:

$$\mathbf{e}^{(1)} = \mathbf{e}, \qquad f_j^{(1)} = \begin{cases} [\bar{x}_i], & \text{ha } j = i, \\ f_j & \text{különben,} \end{cases}$$

$$e_j^{(2)} = \begin{cases} [\bar{x}_i] + 1, & \text{ha } j = i, \\ e_j & \text{különben,} \end{cases} \qquad \mathbf{f}^{(2)} = \mathbf{f}\,.$$

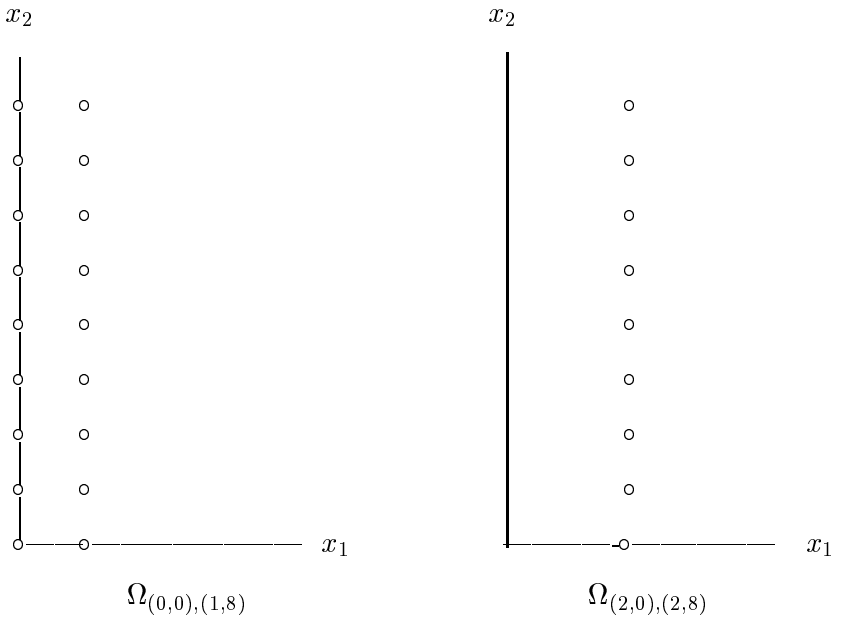
Ekkor $\mathbf{0} \leq \mathbf{e}^{(t)} \leq \mathbf{f}^{(t)} \leq (k, \ldots, k)$ $(t = 1, 2)$. Most legyen

$$\varphi(\Omega_{\mathbf{e},\mathbf{f}}) = \{\Omega_{\mathbf{e}^{(1)},\mathbf{f}^{(1)}}, \Omega_{\mathbf{e}^{(2)},\mathbf{f}^{(2)}}\}.$$

Mivel $\mathbf{e} \neq \mathbf{f}$, ezért $|\Omega_{\mathbf{e},\mathbf{f}}| > 1$. Másrészt egyszerűen belátható, hogy a fenti definícióval φ az $\Omega_{\mathbf{e},\mathbf{f}}$ halmazhoz egy valódi osztályozását rendeli hozzá. Következésképp, φ kielégíti a 7. fejezetben előírt feltételeket abban az esetben, ha a $B\&B$ fa bármely élő levelére $\mathbf{e} \neq \mathbf{f}$ és az $\Omega_{\mathbf{e},\mathbf{f}}$-hez tartozó (9.4) típusú feladat $\bar{\mathbf{x}}$ optimális megoldása nem egész.

Az utóbbiak viszont teljesülnek, ugyanis $\mathbf{e} = \mathbf{f}$ esetén $|\Omega_{\mathbf{e},\mathbf{f}})| = 1$, és így a tekintett levél az iterációs lépés végén lezárásra kerül, mivel $g(\Omega_{\mathbf{e},\mathbf{f}}) \geq \bar{z}$ teljesül. Másrészt, ha a megfelelő (9.4) típusú feladat $\bar{\mathbf{x}}$ optimális megoldása egész, akkor az iterációs lépés végén $g(\Omega_{\mathbf{e},\mathbf{f}}) = z(\bar{\mathbf{x}}) \geq \bar{z}$ teljesül, és így a tekintett levél ismét lezárásra kerül.

Vizsgáljuk ismét az előzőekben tekintett feladatra $\mathbf{e} = (0, 0)$ és $\mathbf{f} = (2, 8)$ mellett az $\Omega_{\mathbf{e},\mathbf{f}}$ szétválasztását. Tudjuk, hogy a megfelelő folytonos feladat $\bar{\mathbf{x}}$ optimális megoldása : $(3/2, 13/2)$. Ekkor $\mathbf{e}^{(1)} = (0, 0)$, $\mathbf{f}^{(1)} = (1, 8)$, $\mathbf{e}^{(2)} = (2, 0)$, $\mathbf{f}^{(2)} = (2, 8)$. A 9.3. ábra mutatja a megfelelő halmazokat.



9.3. ábra. $\Omega_{\mathbf{e},\mathbf{f}}$ osztályozása.

Faépítési stratégia

A felépítésre kerülő eljárásban ugyanazt a stratégiát fogjuk alkalmazni, mint a 7. fejezetben tárgyalt első eljárásnál, azaz az r-edik iterációs lépésben F_r elemei közül (ezek az aktuális $B\&B$ fa élő levelei) kiválasztunk egy minimális korláttal rendelkező elemet.

Ezek után a 7. fejezet jelöléseit és második eljárását felhasználva, a (9.1) feladat megoldására fogunk felépíteni egy Branch-and-Bound eljárást. A tekintett feladathoz nincs olyan közismert heurisztika, melynek segítségével lehetséges megoldást lehetne meghatározni. Másrészt $\mathbf{0} \in L$, így az eljárás előkészítő részében $\mathbf{x}_0$-ként a $\mathbf{0}$ vektort vesszük fel.

Eljárás

Előkészítő rész

- *1.1. lépés.* Legyen $\mathbf{x}^* = \mathbf{0}$, $\bar{z} = z(\mathbf{x}_0) = \alpha$.
- *1.2. lépés.* Konstruáljuk meg a (9.1) feladathoz a (9.2) folytonos feladatot és oldjuk meg szimplex algoritmussal. (Az Ω-hoz rendelt korlát az optimum értéke.)
- *1.3. lépés.* Ha a kapott $\bar{\mathbf{x}}$ optimális megoldás egész, akkor legyen $\mathbf{x}^* = \bar{\mathbf{x}}$ és $\bar{z} = z(\bar{\mathbf{x}})$. Legyen $r = 0$ és

$$F_0 = \begin{cases} \{\Omega\}, & \text{ha } g(\Omega) < \bar{z}, \\ \emptyset & \text{különben.} \end{cases}$$

Ezt követően folytassuk az eljárást az iterációs eljárásrésszel.

Iterációs rész (r-edik iteráció)

- *2.1. lépés.* Ha $F_r = \emptyset$, akkor vége az eljárásnak; $\mathbf{x}^*$ optimális megoldása a feladatnak, az optimum értéke pedig $\bar{z}$.
- *2.2. lépés.* Ha $F_r \neq \emptyset$, akkor vegyünk az F_r-beli elemek közül egy minimális korláttal rendelkező elemet. Jelölje ezt $\Omega_{\mathbf{e},\mathbf{f}}$. Ekkor az $L \cap \Omega_{\mathbf{e},\mathbf{f}}$ halmazt leíró (9.3) típusú feladathoz tartozó folytonos lineáris

programozási feladat $\bar{\mathbf{x}}$ optimális megoldása nem egész, és $\mathbf{e} \leq \bar{\mathbf{x}} \leq \mathbf{f}$. Most képezzük $\bar{\mathbf{x}}$ első nem egész komponense szerint az $\mathbf{e}^{(1)}$, $\mathbf{f}^{(1)}$,$\mathbf{e}^{(2)}$, $\mathbf{f}^{(2)}$ vektorokat az előzőeknek megfelelően. Ekkor

$$\varphi(\Omega_{\mathbf{e},\mathbf{f}}) = \{\Omega_{\mathbf{e}^{(1)},\mathbf{f}^{(1)}}, \Omega_{\mathbf{e}^{(2)},\mathbf{f}^{(2)}}\}.$$

Az $L \cap \Omega_{\mathbf{e}^{(i)},\mathbf{f}^{(i)}}$ halmazokat leíró (9.3) típusú feladatokhoz tartozó folytonos feladatok megoldásával számítsuk ki a $g(\Omega_{\mathbf{e}^{(i)},\mathbf{f}^{(i)}})$ korlátokat $(i = 1, 2)$. Jelölje az optimális megoldásokat rendre $\mathbf{y}^{(1)}$, $\mathbf{y}^{(2)}$. Legyen $X_r = \{\mathbf{u} : \mathbf{u} \in \{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}\}$ és $\mathbf{u}$ egész$\}$. Ha $X_r \neq \emptyset$, akkor legyen $z_r = \min\{z(\mathbf{u}) : \mathbf{u} \in X_r\}$, $\mathbf{x}^{(r)}$ az X_r halmaz egy olyan eleme, amelyre $z_r = z(\mathbf{x}^{(r)})$, és $z_r < \bar{z}$ esetén $\bar{z}$-nek adjuk a z_r értéket, $\mathbf{x}^*$-ot pedig változtassuk $\mathbf{x}^{(r)}$-re. Ezek után legyen

$$F_{r+1} = \{\Omega' : \Omega' \in (F_r \setminus \Omega_{\mathbf{e},\mathbf{f}}) \cup \{\Omega_{\mathbf{e}^{(1)},\mathbf{f}^{(1)}}, \Omega_{\mathbf{e}^{(2)},\mathbf{f}^{(2)}}\} \ \& \ g(\Omega') < \bar{z}\}.$$

Növeljük r értékét 1-gyel és folytassuk az eljárást a következő iterációs lépéssel.

Az eljárás helyessége következik a 7. fejezetben felépített eljárás helyességéből, valamint az alkalmazott Ω halmazra, g korlátozó függvényre és φ szétválasztási függvényre igazolt tulajdonságokból.

Az eljárás demonstrálására tekintsük ismét az előzőekben vizsgált példát. A feladathoz tartozó folytonos feladat optimális megoldása: $(3/2, 13/2)$, az optimum értéke: $-45/2$. Így az előkészítő rész végén $F_0 = \{\Omega\}$, $g(\Omega) = -45/2$, $\mathbf{x}^* = \mathbf{0}$, $\bar{z} = 0$. Ez úgy interpretálható, hogy a $B\&B$ fának egyetlen élő levele van, az Ω halmaz $-45/2$ korláttal. Az eddigi legjobb lehetséges megoldásunk pedig a $\mathbf{0}$ vektor 0 célfüggvényértékkel.

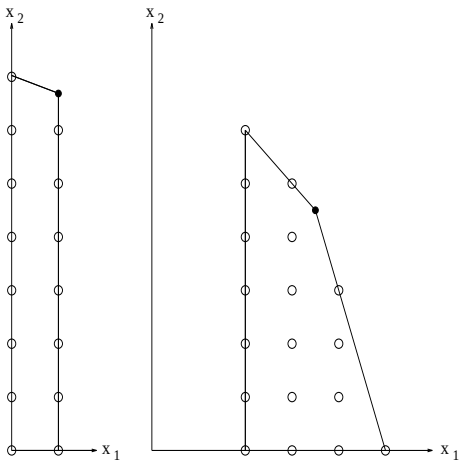
Rátérve az iterációs eljárásrészre, $F_0 \neq \emptyset$. F_0 egyetlen elemet tartalmaz az Ω halmazt. A megfelelő folytonos feladat $(3/2, 13/2)$ optimális megoldásának első nem egész komponense $3/2$. Így $\mathbf{e}^{(1)} = (0, 0)$, $\mathbf{f}^{(1)} = (1, 8)$, $\mathbf{e}^{(2)} = (2, 0)$, $\mathbf{f}^{(2)} = (8, 8)$. Ekkor a szétválasztási függvényünk Ω-hoz hozzárendeli az $\{\Omega_{(0,0),(1,8)}, \Omega_{(2,0),(8,8)}\}$ osztályozást. Az $L \cap \Omega_{(0,0),(1,8)}$ és $L \cap \Omega_{(2,0),(8,8)}$ halmazokat leíró lineáris programozási feladatok a következők:

$$
\begin{array}{l}
x_1 \ +3x_2 \leq 21 \\
x_1 \ +x_2 \ \leq 8 \\
3x_1 \ +x_2 \leq 15 \\
0 \leq x_1 \leq 1 \ , \ x_1 \text{ egész} \\
0 \leq x_2 \leq 8 \ , \ x_2 \text{ egész} \\
\hline
-2x_1 - 3x_2 = z \rightarrow \min
\end{array}
\qquad
\begin{array}{l}
x_1 \ +3x_2 \leq 21 \\
x_1 \ +x_2 \ \leq 8 \\
3x_1 \ +x_2 \leq 15 \\
2 \leq x_1 \leq 8 \ , \ x_1 \text{ egész} \\
0 \leq x_2 \leq 8 \ , \ x_2 \text{ egész} \\
\hline
-2x_1 - 3x_2 = z \rightarrow \min
\end{array}
$$

A fenti feladatokból elhagyva a változókra vonatkozó egész kikötést, a következő folytonos lineáris programozási feladatokat kapjuk:

$$
\begin{array}{l}
x_1 \ +3x_2 \leq 21 \\
x_1 \ +x_2 \ \leq 8 \\
3x_1 \ +x_2 \leq 15 \\
0 \leq x_1 \leq 1 \\
0 \leq x_2 \leq 8 \\
\hline
-2x_1 - 3x_2 = z \rightarrow \min
\end{array}
\qquad
\begin{array}{l}
x_1 \ +3x_2 \leq 21 \\
x_1 \ +x_2 \ \leq 8 \\
3x_1 \ +x_2 \leq 15 \\
2 \leq x_1 \leq 8 \\
0 \leq x_2 \leq 8 \\
\hline
-2x_1 - 3x_2 = z \rightarrow \min
\end{array}
$$

A 9.4. ábra a lehetséges megoldások halmazait mutatja. A sokszögek a folytonos feladatok lehetséges megoldásait írják le, a diszkrét feladatok lehetséges megoldásait pedig karikák jelölik.



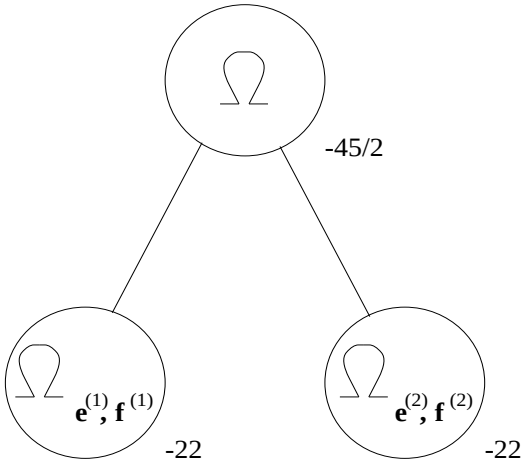
9.4. ábra. A lehetséges megoldások halmazai.

Megoldva a folytonos feladatokat szimplex módszerrel, az első feladat optimális megoldása $\mathbf{y}^{(1)} = (1, 20/3)$, az optimum értéke -22, a második feladat optimális megoldása $\mathbf{y}^{(2)} = (2, 6)$, az optimum értéke -22. Így

$$g(\Omega_{(0,0),(1,8)}) = -22 \qquad \text{és} \qquad g(\Omega_{(2,0),(8,8)}) = -22\,.$$

A kapott két optimális megoldás közül $\mathbf{y}^{(2)}$ egész, így $X_0 = \{\mathbf{y}^{(2)}\}$ és $z_0 = -22$. Mivel z_0 kisebb, mint az eddigi legjobb $\bar{z} = 0$ célfüggvényérték, ezért $\mathbf{x}^*$ $(2, 6)$-ra változik, és $\bar{z}$ felveszi a -22 értéket. De akkor $\bar{z} \leq g(\Omega_{(0,0),(1,8)})$ és $\bar{z} \leq g(\Omega_{(2,0),(8,8)})$, viszont így $F_1 = \emptyset$, amivel véget ér az eljárás. A feladat optimális megoldása $(2, 6)$, az optimum értéke -22.

Ha a $B\&B$ fát ábrázoljuk, akkor az eljárás során az alábbi fa kerül meghatározásra.



9.5. ábra. Az előállított B&B fa.

Mielőtt lezárnánk ezt a fejezetet, vizsgáljuk meg, hogy miként lehet az

$$\begin{gathered}\mathbf{Ax} \leq \mathbf{b} \\ \mathbf{e} \leq \mathbf{x} \leq \mathbf{f} \\ \hline \mathbf{cx} \to \max\end{gathered}$$

feladatokat viszonylag egyszerűen megoldani. Az $\mathbf{e} = \mathbf{0}$ esetben a baloldalon bevezetve nemnegatív eltérésváltozókat, egy lehetséges kanonikus alakú feladatot kapunk, amit szimplex algoritmussal meg tudunk oldani. Ha az

$\bar{\mathbf{x}}$ optimális megoldásban $\bar{x}_i$ nem egész, akkor a szétválasztással kapott $L \cap \Omega_{\mathbf{e}^{(1)},\mathbf{f}^{(1)}}$ halmazt leíró feladat relaxációja egyetlen feltétellel bővül. Az új feltétel: $0 \leq x_i \leq [\bar{x}_i]$. Ezt hozzávéve a szimplex algoritmus végrehajtásával kapott utolsó lehetséges kanonikus alakú feladat feltételrendszeréhez, majd a baloldalát bővítve egy nemnegatív eltérésváltozóval, majdnem lehetséges kanonikus alakú feladatot kapunk. A probléma csupán az, hogy az x_i bázisváltozó szerepel az új egyenletben is. Most kivonva az x_i bázisváltozó egyenletét az új egyenletből, az x_i változó már csak egy egyenletben fog szerepelni. A kivonással viszont elromlik az új egyenlet jobboldala, ami $[\bar{x}_i] - \bar{x}_i$ lesz. Így, mivel $\bar{x}_i$ nem egész, az új egyenlet jobboldala negatívvá válik. Most vegyük észre, hogy az előálló feladat pontosan olyan feladat, amire alkalmazható a duális szimplex algoritmus.

A szétválasztással kapott $L \cap \Omega_{\mathbf{e}^{(2)},\mathbf{f}^{(2)}}$ halmazt leíró feladat relaxációja szintén egyetlen feltétellel bővül. Az új feltétel: $[\bar{x}_i]+1 \leq x_i$. Ezt megszorozva -1-gyel és hozzávéve a szimplex algoritmus végrehajtásával kapott utolsó lehetséges kanonikus alakú feladat feltételrendszeréhez, majd a baloldalát bővítve egy nemnegatív eltérésváltozóval, ismét olyan feladatot kapunk, ami nagyon hasonlít a lehetséges kanonikus alakú feladatra. Az egyik gond, hogy az x_i báziváltozó szerepel az új egyenletben negatív előjellel. Ezt kiküszöbölhetjük úgy, hogy az x_i bázisváltozó egyenletét hozzáadjuk az új egyenlethez. Ekkor az új egyenlet jobboldala: $\bar{x}_i - [\bar{x}_i] - 1$, ami negatív. De ez nem probléma, mert ismét olyan feladathoz jutottunk, amely megoldható duális szimplex algoritmussal.

Egyszerűen átgondolható, hogy a fentiekben leírt megoldási technika az eljárásban előforduló bármely $\mathbf{e}$, $\mathbf{f}$ párosnál alkalmazott szétválasztásnál használható.

A jelen fejezetben ismertetett módszerrel kapcsolatosan meg kell jegyezni, hogy az eljárás nem csak (9.1) típusú feladatok megoldására alkalmas, hanem egy sokkal általánosabb feladatosztály problémáira, nevezetesen a vegyes egészértékű lineáris programozási feladatok megoldására is alkalmazható. Az általánosabb hatókörű eljárás magyar nyelvű tárgyalása megtalálható Krekó Béla [114] könyvében.

A fejezethez kapcsolódva megemlítjük még, hogy az egészértékű programozási feladatok megoldására szolgáló technikák témaköre, az egészértékű programozás matematikai háttere igen intenzíven kutatott terület. Számos bevezető jellegű könyv nyert publikálást ebből a témakörből, melyek közül a teljesség igénye nélkül felsorolunk néhányat. Igen jó, az alapokat tárgyaló munkáknak számítanak a [70], [159], [169] könyvek.

10. Utazó ügynök probléma

A modellhez számos gyakorlati probléma kapcsolható. Ezek közül a legismertebb a címben szereplő alábbi feladat.

Az utazó ügynök problémája

Adott n számú város és a városokat összekötő utak, amelyeknek ismert a hossza. Adott továbbá egy ügynök, akinek adott városból kiindulva, minden várost végig kell látogatnia úgy, hogy minden várost pontosan egyszer érint, és az út befejeztével visszatér a kiindulási városba. Határozzuk meg az ügynök legrövidebb útját.

Vezessük be a következő jelöléseket.

Jelölje $1, 2, \ldots, n$ a városokat és c_{ij} az i-edik és j-edik városokat összekötő út hosszát. Ha a két várost nem köti össze út, akkor legyen $c_{ij} = W$, ahol W már az előzőekben is alkalmazott megfelelően nagy szám.

Legyen

$$x_{ij} = \begin{cases} 1, & \text{ha az ügynök az } i\text{-edik városból a } j\text{-edik városba megy,} \\ 0 & \text{különben.} \end{cases}$$

Végül jelölje tetszőleges $Q \subseteq \{1, \ldots, n\}$ halmazra $\bar{Q}$ az $\{1, \ldots, n\} \setminus Q$ halmazt. Akkor a fenti feladat az alábbi, *utazó ügynök probléma* néven ismert optimumszámítási modellel írható le.

$$\begin{gathered} \sum_{t=1}^{n} x_{it} = 1 \ \ (i = 1, \ldots, n) \\ \sum_{t=1}^{n} x_{tj} = 1 \ \ (j = 1, \ldots, n) \qquad (10.1) \\ \sum_{i \in Q} \sum_{j \in \bar{Q}} x_{ij} \geq 1 \ \ (\emptyset \neq Q \subset \{1, \ldots, n\}) \\ x_{ij} \in \{0, 1\}, \ \ (i = 1, \ldots, n; j = 1, \ldots, n) \\ \hline \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} = z \to \min \end{gathered}$$

Az első feltételcsoport garantálja, hogy az ügynök minden városból távozik. A második feltételcsoport biztosítja, hogy az ügynök eljut minden városba. Sajnálatos módon a két feltételcsoport megengedi diszjunkt részkörutak kialakulását. Ezt küszöböli ki a harmadik feltételcsoport, amelyben előírjuk, hogy a városok tetszőleges $\emptyset \neq Q \subset \{1, \ldots, n\}$ részhalmazára az ügynök valamely Q-beli városból egy nem Q-beli városba látogat. Ez valóban kizárja a részkörutat, ugyanis a részkörútban szereplő városok halmazát tekintve Q-nak, a harmadik feltételcsoport megfelelő feltétele nem teljesül.

Megjegyezzük, hogy az utazó ügynök problémája a korábbi fejezetek terminológiájával összhangban úgy interpretálható, hogy egy minimális hosszúságú, minden csúcspontot tartalmazó irányított körutat kell meghatározni egy adott hálózatban.

Az utazó ügynök problémáján kívül számos olyan gyakorlati feladat van, amely a (10.1) modellre vezethető vissza. A továbbiakban két ilyen gyakorlati alkalmazást ismertetünk.

Műszaki tervezés

Adott egy automata gépsor és abban egy szegecselő gép. A szegecselő gép karjának a szalagon jövő termék meghatározott pontjaiba szegecseket kell elhelyeznie. Határozzuk meg a szegecsek elhelyezésének azt a sorrendjét, amely esetén legkisebb a szegecselő gép karjának teljes mozgása.

Sorrendi ütemezés

Adott egy üzem, amely n féle különböző terméket gyárt. Az egyes termékféleségek gyártása időben elkülönül egymástól, és a termékváltásnál a gépeket át kell állítani. Ismeretes tetszőleges termékpárra a gépek átállításának a költsége. Határozzuk meg a termékek termelésének egy olyan sorrendjét, amely mellett a gépek átállításának teljes költsége minimális.

A (10.1) modellel kapcsolatban vegyük észre, hogy a harmadik feltételcsoportban a feltételek száma $2^n - 2$, ami már viszonylag kicsi n mellett is nagyon sok egyenlőtlenséget eredményez. A feltételek számának csökkentésére a problémának több ekvivalens formalizálása is kidolgozásra került. A következő (10.2) változatot A. Tucker [171] dolgozta ki 1960-ban.

$$\sum_{t=1}^{n} x_{it} = 1 \ \ (i = 1, \ldots, n)$$

$$\sum_{t=1}^{n} x_{tj} = 1 \ \ (j = 1, \ldots, n) \tag{10.2}$$

$$u_i - u_j + (n-1)x_{ij} \leq n - 2 \ \ (2 \leq i \neq j \leq n)$$

$$x_{ii} = 0, \ \ x_{ij} \in \{0, 1\}, \ \ (i = 1, \ldots, n; j = 1, \ldots, n)$$

$$u_i \geq 0 \ \ \& \ \text{egész} \ \ (i = 2, \ldots, n)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} = z \to \min$$

Az ekvivalencia igazolásához egyrészt azt kell megmutatnunk, hogy diszjunkt részkörutak egyesítése nem elégíti ki a (10.2) feladat feltételrendszerét. Ezt indirekt bizonyítjuk. Tegyük fel, hogy a feladat valamely $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ lehetséges megoldására

$$\bar{x}_{i_1 i_2} = \bar{x}_{i_2 i_3} = \ldots \bar{x}_{i_k i_1} = 1$$

teljesül valamely $1 < k < n$ egészre. Az általánosság megszorítása nélkül feltehetjük, hogy $1 \notin \{i_1, \ldots, i_k\}$. Mivel $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ lehetséges megoldás, ezért

$$\bar{u}_{i_1} - \bar{u}_{i_2} + (n-1)\bar{x}_{i_1 i_2} \leq n - 2$$

$$\bar{u}_{i_2} - \bar{u}_{i_3} + (n-1)\bar{x}_{i_2 i_3} \leq n - 2$$

$$\vdots$$

$$\bar{u}_{i_k} - \bar{u}_{i_1} + (n-1)\bar{x}_{i_k i_1} \leq n - 2$$

teljesül. Összeadva rendre az egyenlőtlenségek bal- és jobboldalait, az alábbi egyenlőtlenségnek kell fennállnia

$$k(n-1) \leq k(n-2) \, ,$$

ami $1 < k < n$ esetén ellentmondás.

Másrészt igazolnunk kell, hogy bármely körúthoz létezik a (10.2) feladatnak egy olyan $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ lehetséges megoldása, hogy az $\bar{\mathbf{X}}$ által meghatározott élek pontosan a tekintett körutat szolgáltatják. Ehhez tekintsünk egy tetszőleges, az $(1, i_2), (i_2, i_3), \ldots, (i_n, 1)$ éleket tartalmazó körutat. Definiáljuk $\bar{\mathbf{X}}$-t és $\bar{\mathbf{u}}$-t a következők szerint:

$$\bar{x}_{1i_2} = \bar{x}_{i_2 i_3} = \ldots \bar{x}_{i_n 1} = 1 \,,$$

$$\bar{x}_{ij} = 0 \text{ a többi indexpárra,}$$

$$\bar{u}_{i_t} = t \quad (t = 2, \ldots, n) \,.$$

Megmutatjuk, hogy $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ kielégíti (10.2) feltételrendszerét. A feltételek teljesülése a harmadik feltételcsoporttól eltekintve nyilvánvaló. Legyen ezek után $2 \leq i \neq j \leq n$ tetszőleges indexpár. Az $\bar{\mathbf{u}}$ vektor definíciójából következik, hogy $\bar{u}_i - \bar{u}_j \leq n - 2$. Másrészt $\bar{x}_{ij} = 1$ akkor és csak akkor teljesül, ha $i = i_r$ és $j = i_{r+1}$ valamely $2 \leq r \leq n$ indexre. De ebben az esetben

$$\bar{u}_i - \bar{u}_j + (n-1)\bar{x}_{ij} = r - (r+1) + (n-1) \leq n - 2 \,,$$

azaz a harmadik feltételcsoport feltételei rendre teljesülnek, így $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ lehetséges megoldása (10.2)-nek.

A vizsgált modellel kapcsolatban fontos megjegyezni, hogy $\mathbf{C}$-re vonatkozóan semmiféle kikötést nem tartalmaz. Ez a legáltalánosabb modellje az utazó ügynök problémának, és a jelen fejezetben ennek vizsgálatára szorítkozunk. A következő fejezetben speciális változatokkal is megismerkedünk. Például vizsgáljuk azt a modellt, amelyben $c_{ij} = c_{ji}$ $(1 \leq i \leq n; 1 \leq j \leq n)$ tejesül, azaz a költségmátrix szimmetrikus. A szimmetria bizonyos alkalmazásokban teljesül. A fentiekben megadott alkalmazásokban, az ügynök mozgásánál, a szegecselő gép karjának mozgásánál feltételezhető a szimmetria, míg a sorrendi ütemezésnél már nem feltétlenül érvényes. (Az utóbbi alkalmazásnál ez azt jelentené, hogy az i-edik termékről a j-edik termék gyártására történő átállás ugyanannyi költséggel jár, mint a j-edik termékről az i-edik termék gyártására történő átállás, ami általában nem igaz.)

Sajnos az utazó ügynök probléma különböző formalizálásai nem eredményeztek előrelépést a probléma megoldásában. 1976-ban bizonyítást nyert a [67] dolgozatban, hogy az általános utazó ügynök probléma az NP-teljes

feladatok osztályához tartozik, így $P \neq NP$ esetén nem létezik hatékony (az n polinomjával korlátozható időigényű) eljárás a feladat megoldására. Ezért különböző Branch-and-Bound eljárások kerültek kidolgozásra az optimális körút meghatározására, és számos heurisztikus algoritmust készítettek közelítő optimumértéket biztosító lehetséges megoldások meghatározására. Az ilyen megoldásokat szokásos *szuboptimális megoldásnak* is nevezni.

A továbbiakban mi is felépítünk egy Branch-and-Bound eljárást, majd ezt követően néhány heurisztikus eljárást ismertetünk a következő fejezetben. Mindezekhez szükségesek bizonyos előkészületek.

Vizsgáljuk ezek után a (10.1) feladatot. Az általánosság megszorítása nélkül feltehetjük, hogy az ügynök az 1-gyel jelzett városból indul. Ekkor $n-1$ számú városba mehet, majd $n-2$-be, és így tovább. Ebből az következik, hogy a lehetséges megoldások száma $(n-1)!$. Viszont így rögtön adódik, hogy a (10.1) feladatnak mindig létezik optimális megoldása. Ezzel kapcsolatban célszerű megjegyezni, hogy amennyiben az i-edik városból nem vezetett út a j-edik városba, akkor ezt a modellben egy W hosszúságú fiktív úttal helyettesítettük. Így, ha optimumértékként W adódik, akkor ez azt jelenti, hogy a városok közötti úthálózat olyan, amely nem teszi lehetővé a körbejárást, azaz a gyakorlati problémának nincs lehetséges megoldása.

Vegyük észre, hogy a (10.1) feladat feltételrendszere csak n-től függ. Így a $\mathbf{C}^{n\times n}$ költségmátrix egyértelműen meghatározza a feladatot. Ennek alapján a (10.1) feladatra a $TSP(\mathbf{C})$ jelöléssel fogunk hivatkozni. Itt az angol *Traveling Salesman Problem* név rövidítését használjuk, mely a témakör irodalmában igen széles körben elterjedt.

A továbbiakban az egyszerűbb tárgyalás érdekében az $\bar{\mathbf{X}}$ *mátrixot körútnak* nevezzük, ha minden $\bar{x}_{ij} = 1$-re véve az (i,j) élet, az így képezett élek az n-szögpontú teljes gráfban irányított kört alkotnak.

Most ismételten használjuk a 3.2. fejezetben bevezetett mátrixok ekvivalenciáját.

A tekintett reláció és a TSP kapcsolatát adja meg a következő állítás.

10.1. segédtétel. *Ha* $\mathbf{C}^{n\times n} \sim \mathbf{D}^{n\times n}$, *akkor a* $TSP(\mathbf{C})$ *és* $TSP(\mathbf{D})$ *feladatok optimális megoldásai megegyeznek.*

Bizonyítás. Mivel mindkét feladatnak létezik optimális megoldása, ezért az állítás korrekt. Másrészt a tekintett két feladat feltételrendszere azonos, ezért a lehetséges megoldások halmaza közös, amelyet jelöljön L. Továbbá

jelölje a $TSP(\mathbf{C})$ és $TSP(\mathbf{D})$ feladatok célfüggvényét rendre $z_{\mathbf{C}}$ és $z_{\mathbf{D}}$. Megmutatjuk, hogy van olyan γ konstans, amelyre $z_{\mathbf{C}}(\bar{\mathbf{X}}) = z_{\mathbf{D}}(\bar{\mathbf{X}}) + \gamma$ teljesül tetszőleges $\bar{\mathbf{X}}$ lehetséges megoldásra. Valóban, mivel $\mathbf{C} \sim \mathbf{D}$, ezért vannak olyan $\alpha_1, \ldots, \alpha_n$; $\beta_1, \ldots, \beta_n$ konstansok, hogy $c_{ij} = d_{ij} + \alpha_i + \beta_j$ teljesül bármely $1 \leq i \leq n$, $1 \leq j \leq n$ indexpárra. De akkor

$$z_{\mathbf{C}}(\bar{\mathbf{X}}) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\bar{x}_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n}(d_{ij} + \alpha_i + \beta_j)\bar{x}_{ij} =$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\bar{x}_{ij} + \sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}\bar{x}_{ij} + \sum_{j=1}^{n}\beta_j\sum_{i=1}^{n}\bar{x}_{ij} \; .$$

Mivel $\bar{\mathbf{X}}$ lehetséges megoldás, ezért $\sum_{j=1}^{n}\bar{x}_{ij} = 1$ és $\sum_{i=1}^{n}\bar{x}_{ij} = 1$. De akkor

$$z_{\mathbf{C}}(\bar{\mathbf{X}}) = z_{\mathbf{D}}(\bar{\mathbf{X}}) + \sum_{i=1}^{n}\alpha_i + \sum_{j=1}^{n}\beta_j \; .$$

Most legyen $\gamma = \sum_{i=1}^{n}\alpha_i + \sum_{j=1}^{n}\beta_j$. Akkor a $z_{\mathbf{C}}(\bar{\mathbf{X}}) = z_{\mathbf{D}}(\bar{\mathbf{X}}) + \gamma$ egyenlethez jutunk, amely teljesül bármely $\bar{\mathbf{X}}$ lehetséges megoldásra. Ez pontosan azt jelenti, hogy a lehetséges megoldások halmazán a két célfüggvény csak egy additív konstansban tér el egymástól, amiből nyilvánvalóan következik az állítás.

10.1. következmény. *Az optimális megoldás meghatározását illetően elegendő olyan $TSP(\mathbf{C})$ feladatok vizsgálatára szorítkoznunk, amelyekre $\mathbf{C} \geq 0$ teljesül.*

A fenti következmény alapján a továbbiakban minden hivatkozás nélkül feltételezzük, hogy a vizsgált feladatok célfüggvényegyütthatói rendre nemnegatívak.

A (10.1) feladattal kapcsolatosan azt is észrevehetjük, hogy amennyiben a harmadik feltételcsoporttól eltekintünk, úgy a 3.2. fejezetben megismert hozzárendelési feladathoz jutunk. Ebből viszont következik, hogy (10.1) tetszőleges $\bar{\mathbf{X}}$ lehetséges megoldása egyben lehetséges megoldása a $\mathbf{C}$ költségmátrixú $A(\mathbf{C})$ hozzárendelési feladatnak is. Jelölje (10.1) lehetséges megoldásainak halmazát L, $A(\mathbf{C})$ lehetséges megoldásainak halmazát S. Akkor $L \subseteq S$, és így

$$\min\{z(\mathbf{X}) : \mathbf{X} \in S\} \leq \min\{z(\mathbf{X}) : \mathbf{X} \in L\} \; .$$

A kapott egyenlőtlenségből nyilvánvalóan adódnak a következők:

(i) *ha* $\bar{\mathbf{X}}$ *optimális megoldása* $A(\mathbf{C})$*-nek és* $\bar{\mathbf{X}}$ *körút, akkor* $\bar{\mathbf{X}}$ *egyben optimális megoldása* $TSP(\mathbf{C})$*-nek is,*

(ii) *ha* $\bar{\mathbf{X}}$ *optimális megoldása* $A(\mathbf{C})$*-nek, akkor* $z(\bar{\mathbf{X}})$ *alsó korlátja a* $TSP(\mathbf{C})$ *feladat optimumértékének.*

A továbbiakban olyan speciális hozzárendelési feladatokat használunk, melyekre kikötjük, hogy bizonyos változóknak 0 értéket, a változók egy másik csoportjának 1 értéket kell felvennie. Ismét I és J jelöli azon változók indexeinek a halmazát, amelyekről kikötjük, hogy értékük 1 illetve 0. Ezen speciális hozzárendelési feladatok jelölésére az $(A(\mathbf{C}), I, J)$ jelölést fogjuk használni.

A $TSP(\mathbf{C})$ és $A(\mathbf{C})$ feladatok kapcsolatának demonstrálására tekintsük a következő feladatokat.

10.1. példa

Vizsgáljuk az alábbi költségmátrixú 6 városos $TSP(\mathbf{C})$ feladatot, ahol az $i \to i$ típusú utak kizárására a c_{ii} együtthatókat rendre W-nek választottuk. (W-ként például használhatjuk az $1 + n \cdot \max\{c_{ij} : 1 \le i \le n; 1 \le j \le n, i \ne j\}$ értéket is.)

	1	2	3	4	5	6
1	W	6	11	6	16	19
2	5	W	16	13	21	9
3	8	12	W	22	18	11
4	20	18	27	W	22	15
5	10	22	11	25	W	18
6	16	10	17	30	10	W

Oldjuk meg a megfelelő $A(\mathbf{C})$ hozzárendelési feladatot a 3.2. fejezetben megismert magyar módszerrel.

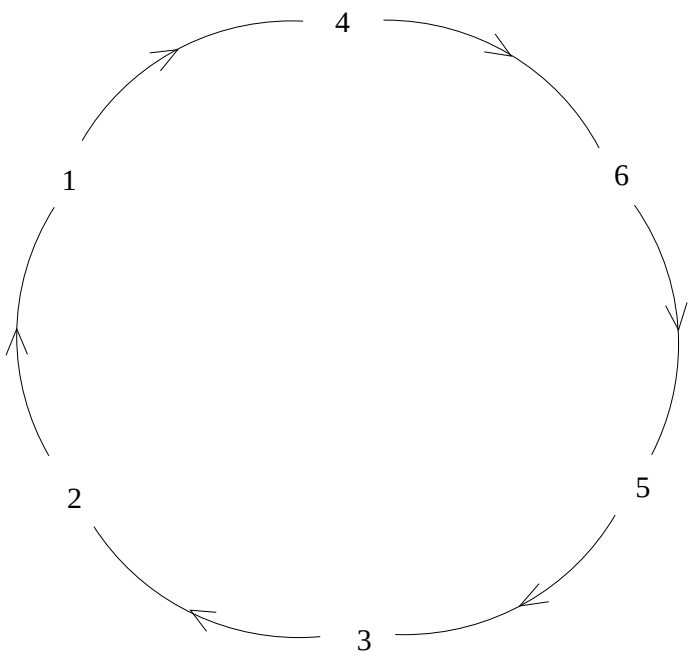
A magyar módszer által szolgáltatott utolsó mátrix a következő:

$$\begin{array}{cccccc} W & 0 & 8 & 0^* & 10 & 16 \\ 0^* & W & 10 & 4 & 12 & 3 \\ 0 & 0^* & W & 10 & 6 & 2 \\ 6 & 0 & 12 & W & 4 & 0^* \\ 0 & 8 & 0^* & 11 & W & 7 \\ 10 & 0 & 10 & 20 & 0^* & W \end{array}$$

Így az $A(\mathbf{C})$ hozzárendelési feladat optimális megoldása az alábbi módon definiált $\bar{\mathbf{X}}$ mátrix:

$\bar{x}_{14} = 1$, $\bar{x}_{21} = 1$, $\bar{x}_{32} = 1$, $\bar{x}_{46} = 1$, $\bar{x}_{53} = 1$, $\bar{x}_{65} = 1$ és $\bar{x}_{ij} = 0$ a többi indexpárra.

A megfelelő élek az alábbi körutat eredményezik:



10.1. ábra. A 10.1. példa optimális megoldása.

Következésképp, $\bar{\mathbf{X}}$ a $TSP(\mathbf{C})$ feladatnak is optimális megoldása. Az optimum értéke:

$$z(\bar{\mathbf{X}}) = c_{14} + c_{21} + c_{32} + c_{46} + c_{53} + c_{65} = 59 \ .$$

Sajnos a megfelelő hozzárendelési feladat optimális megoldása nem minden esetben eredményez körutat. Ez a helyzet az alábbi E. Balastól származó, a [123] munkában demonstrációs célokra használt feladat esetén is.

10.2. példa

Tekintsük a következő költségmátrixú 8 városos TSP feladatot.

$$\begin{pmatrix} W & 2 & 11 & 10 & 8 & 7 & 6 & 5 \\ 6 & W & 1 & 8 & 8 & 4 & 6 & 7 \\ 5 & 12 & W & 11 & 8 & 12 & 3 & 11 \\ 11 & 9 & 10 & W & 1 & 9 & 8 & 10 \\ 11 & 11 & 9 & 4 & W & 2 & 10 & 9 \\ 12 & 8 & 5 & 2 & 11 & W & 11 & 9 \\ 10 & 11 & 12 & 10 & 9 & 12 & W & 3 \\ 7 & 10 & 10 & 10 & 6 & 3 & 1 & W \end{pmatrix}$$

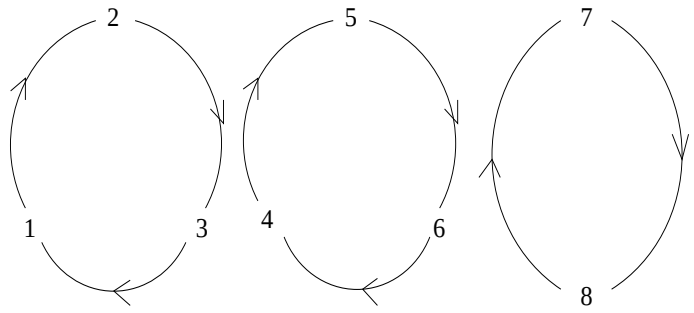
Rendre kivonva a 2, 1, 3, 1, 2, 2, 3, 1 sorminimumokat, majd az előálló $\bar{\mathbf{C}}$ mátrix első oszlopára a 2 oszlopminimumot, a kapott $\mathbf{C}^{(0)}$ mátrixban oszlopfolytonosan kijelölve a független 0-rendszert, a következő mátrixhoz jutunk:

$$\begin{pmatrix} W & 0^* & 9 & 8 & 6 & 5 & 4 & 3 \\ 3 & W & 0^* & 7 & 7 & 3 & 5 & 6 \\ 0^* & 9 & W & 8 & 5 & 9 & 0 & 8 \\ 8 & 8 & 9 & W & 0^* & 8 & 7 & 9 \\ 7 & 9 & 7 & 2 & W & 0^* & 8 & 7 \\ 8 & 6 & 3 & 0^* & 9 & W & 9 & 7 \\ 5 & 8 & 9 & 7 & 6 & 9 & W & 0^* \\ 4 & 9 & 9 & 9 & 5 & 2 & 0^* & W \end{pmatrix}$$

A kapott független 0-rendszer n-elemű, és így a hozzárendelési feladatra a következő optimális megoldást kapjuk: $\bar{x}_{12} = \bar{x}_{23} = \bar{x}_{31} = \bar{x}_{45} = \bar{x}_{56} = \bar{x}_{64} = \bar{x}_{78} = \bar{x}_{87} = 1$ és $\bar{x}_{ij} = 0$ a többi indexpárra.

$$z(\bar{\mathbf{X}}) = c_{12} + c_{23} + c_{31} + c_{45} + c_{56} + c_{64} + c_{78} + c_{87} = 17$$

A megfelelő élek a 10.2. ábrán megadott részkörutakat eredményezik.



10.2. ábra. A hozzárendelési feladat megoldásával adódó részkörutak.

Annak ellenére, hogy a hozzárendelési feladat optimális megoldása nem eredményezett körutat, bizonyos információkat nyerhetünk $A(\mathbf{C})$ optimumából. Nevezetesen, (ii) alapján $z(\bar{\mathbf{X}}) = 17$ alsó korlátja az optimumértéknek.

Ezen korlát élességét illetően igen érdekes E. Balas és P. Toth [123] számítógépes kísérlete. Az említett kutatók 400 problémát generáltak a vizsgálathoz. Minden feladatnál egyenletes eloszlás mellett véletlenszerűen választottak feladatméretet az $50 \leq n \leq 250$ tartományból, valamint célfüggvényegyütthatókat az 1 és 100 vagy az 1 és 1000 közé eső egészekből. Az ilyen módon előállított feladatokra rendre meghatározták az optimális körút költségét és a megfelelő hozzárendelési feladat optimumát. A kapott értékekre képezve az átlagokat, azt kapták, hogy a hozzárendelési feladatok optimumainak az átlaga 99.2%-a az utazó ügynök problémák optimumai átlagának. A vizsgálati adatok egy olyan tendenciát is mutattak, hogy n növelésével egyre élesebbé válik a korlát.

A következőkben egy $B\&B$ eljárást építünk fel az utazó ügynök probléma megoldására, amelyben a korlátozó függvény definiálására a fentiekben vázolt, az $A(\mathbf{C})$ és $TSP(\mathbf{C})$ feladatok közti kapcsolatot fogjuk kihasználni. Az eljárás felépítését a 7. fejezetnek megfelelően végezzük. Feltételezzük, hogy a $\mathbf{C}$ mátrixban $c_{ii} = W$, $(i = 1, \ldots, n)$ teljesül.

I. Az Ω halmaz megadása

Legyen Ω a $\{0, 1\}$ feletti $n \times n$-es mátrixok halmaza. Nyilvánvaló, hogy $L \subseteq \Omega$, továbbá $|\Omega| = 2^{n^2}$, így Ω rendelkezik a kívánt tulajdonságokkal.

II. A φ szétválasztási és a g korlátozó függvények megadása

Elsőként a függvényeket az Ω halmazon definiáljuk. Legyen $g(\Omega)$ az

$A(\mathbf{C})$ hozzárendelési feladat optimumértéke. A $\varphi(\Omega)$ definiálásához pedig különböztessük meg az alábbi két esetet.

(a) Ha $A(\mathbf{C})$ optimális megoldása körút, akkor ez optimális megoldása a $TSP(\mathbf{C})$ feladatnak is. Így $\bar{z}$ felveszi az optimumértéket és Ω nem lesz élő levél, azaz $\Omega \notin F_0$. De akkor nem kell a φ függvényt az Ω halmazon definiálni.

(b) Ha az $A(\mathbf{C})$ feladat $\bar{\mathbf{X}}$ optimális megoldása nem körút, akkor $\bar{\mathbf{X}}$ diszjunkt részkörökből áll. Legyen ezek közül az $(i_1, i_2), \ldots, (i_{k-1}, i_k), (i_k, i_1)$ éleket tartalmazó részkörút minimális elemszámú, és tekintsük a következő halmazokat:

$$\begin{aligned}
\Omega^{(0)} &= \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{i_1 i_2} = \ldots = x_{i_k i_1} = 1\},\\
\Omega^{(1)} &= \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{i_1 i_2} = 0\},\\
\Omega^{(2)} &= \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{i_2 i_3} = 0 \ \& \ x_{i_1 i_2} = 1\},\\
&\vdots\\
\Omega^{(k)} &= \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{i_k i_1} = 0 \ \& \ x_{i_1 i_2} = \ldots = x_{i_{k-1} i_k} = 1\}.
\end{aligned}$$

Egyszerűen beláthatók a következők:

(1) az $\Omega^{(0)}, \Omega^{(1)}, \ldots, \Omega^{(k)}$ halmazok a felbontandó halmaznak (jelenleg Ω) egy valódi osztályozását alkotják,

(2) bármely $1 \leq i \leq k$ indexre az $\Omega^{(i)}$ definíciójában 1 értékkel szereplő változóknak megfelelő élekből álló gráf nem tartalmaz részkörutat,

(3) $\Omega^{(0)} \cap L = \emptyset$.

Definiáljuk $\varphi(\Omega)$-t a következők szerint. Legyen

$$\varphi(\Omega) = \{\Omega^{(0)}, \Omega^{(1)}, \ldots, \Omega^{(k)}\} .$$

A φ függvény definícióját úgy fogjuk kiterjeszteni, hogy az (1),(2),(3) tulajdonságok érvényben maradjanak. A továbbiakban az osztályozások osztályainak leírására azt a technikát fogjuk használni, hogy az I, J halmazokban megadjuk azon változók indexeit, amelyek értéke rögzített. Például $\Omega^{(2)} = \Omega_{I,J}$, ahol $I = \{(i_1, i_2)\}$, $J = \{(i_2, i_3)\}$. A rögzített értékű változókat *kötött változóknak*, a többi változót pedig *szabad változóknak* fogjuk nevezni $\Omega_{I,J}$-re vonatkozóan. Végül speciálisan azokat az osztályokat, amelyekről tudjuk, hogy nem tartalmaznak körutat (Ω felbontásánál $\Omega^{(0)}$), lássuk el $*$-gal.

Ezek után kiterjesztjük a φ és g függvények értelmezését.

A g korlátozó függvény definiálásához tekintsük a $B\&B$-fában valamely élő levél leszármazottját. Ha a tekintett osztály $*$-gal lett ellátva, akkor g

rendelje ehhez a részhalmazhoz a W korlátot. Ellenkező esetben jelölje a tekintett osztályt $\Omega_{I,J}$. Ekkor az $(A(\mathbf{C}), I, J)$ hozzárendelési feladat bármely lehetséges megoldása eleme $\Omega_{I,J}$-nek, továbbá bármely olyan körút, amelyben az I-ben szereplő indexekre a változók értéke 1, valamint a J-ben szereplő indexekre a változók értéke 0, lehetséges megoldása $(A(\mathbf{C}), I, J)$-nek. Viszont így az $(A(\mathbf{C}), I, J)$ feladat $\tilde{z}$ optimumértékére

$$\tilde{z} \leq \min\{z(\mathbf{X}) : \mathbf{X} \in L \cap \Omega_{I,J}\}$$

teljesül, amennyiben $L \cap \Omega_{I,J} \neq \emptyset$. Ennek alapján definiáljuk $g(\Omega_{I,J})$-t a következők szerint:

$$g(\Omega_{I,J}) = \begin{cases} (A(\mathbf{C}), I, J) \text{ optimuma}, & \text{ha } |\Omega_{I,J}| > 1, \\ z(\bar{\mathbf{X}}), & \text{ha } \Omega_{I,J} = \{\bar{\mathbf{X}}\} \text{ és } \bar{\mathbf{X}} \text{ körút}, \\ W & \text{különben.} \end{cases}$$

A φ függvény definíciójának kiterjesztéséhez tekintsük a $B\&B$-fa egy élő $\Omega_{I,J}$ levelét. Mivel $\Omega_{I,J}$ élő levél, ezért $|I| + |J| < n^2 - n$, $(A(\mathbf{C}), I, J)$ optimuma kisebb, mint W, továbbá $(A(\mathbf{C}), I, J)$ optimális megoldása nem körút. (Ellenkező esetben a levél lezárásra kerülne.) De akkor $(A(\mathbf{C}), I, J)$ optimális megoldása diszjunkt részkörutak uniója. Válasszunk ezen részkörutak közül egy minimális elemszámút. Jelölje K a választott részkörúthoz tartozó változók indexeinek a halmazát. Akkor K nemüres halmaz, $K \cap J = \emptyset$, és a (2) tulajdonság miatt $K \not\subseteq I$. Legyen $K \setminus I = \{(i_1, j_1), \ldots, (i_r, j_r)\}$. Mivel $(K \setminus I) \cap J = \emptyset$ és $(K \setminus I) \cap I = \emptyset$, ezért bármely $x_{i_s j_s}$ $(1 \leq s \leq r)$ változó szabad változó lesz $\Omega_{I,J}$-re vonatkozóan. Képezzük ezek után a következő halmazokat:

$$\begin{aligned}
\Omega_{I_0,J_0} &= \{\mathbf{X} : \mathbf{X} \in \Omega_{I,J} \ \& \ x_{i_1 j_1} = \ldots = x_{i_r j_r} = 1\}, \\
\Omega_{I_1,J_1} &= \{\mathbf{X} : \mathbf{X} \in \Omega_{I,J} \ \& \ x_{i_1 j_1} = 0\}, \\
\Omega_{I_2,J_2} &= \{\mathbf{X} : \mathbf{X} \in \Omega_{I,J} \ \& \ x_{i_2 j_2} = 0 \ \& \ x_{i_1 j_1} = 1\}, \\
&\vdots \\
\Omega_{I_r,J_r} &= \{\mathbf{X} : \mathbf{X} \in \Omega_{I,J} \ \& \ x_{i_r j_r} = 0 \ \& \ x_{i_1 j_1} = \ldots = x_{i_{r-1} j_{r-1}} = 1\}.
\end{aligned}$$

Mivel $x_{i_t j_t}$ $(1 \leq t \leq r)$ szabad változók $\Omega_{I,J}$-re vonatkozóan, ezért a fenti halmazok az $\Omega_{I,J}$ halmaznak egy valódi osztályozását alkotják. Nyilvánvaló, hogy $\Omega_{I_0,J_0} \cap L = \emptyset$, ugyanis Ω_{I_0,J_0} minden eleme tartalmazza a kiválasztott minimális elemszámú részkörutat, mint részgráfot. Most legyen $0 \leq t \leq r$ tetszőleges egész. Jelölje $\mathcal{G}_t = (N, E_t)$ azt a gráfot, amelyre $(i, j) \in E_t$ akkor és csak akkor teljesül, ha $(i, j) \in I_t$, ahol $N = \{1, \ldots, n\}$.

Ekkor $\mathcal{G}_0$ tartalmazza a kiválasztott minimális elemszámú részkörutat és más részkörutat nem tartalmaz. Másrészt bármely $1 \leq t \leq r$ indexre $\mathcal{G}_t$ előállítható $\mathcal{G}_0$-ból úgy, hogy $\mathcal{G}_0$ egyetlen részkörútjából elhagyunk egy vagy több élet. De akkor $\mathcal{G}_t$ nem tartalmaz részkörutat. Ez pontosan azt jelenti, hogy az Ω_{I_t,J_t}-ben 1 értékkel rögzített változóknak megfelelő élek nem alkotnak részkörutat bármely $1 \leq t \leq r$ indexre. Következésképp, a definiált halmazok rendelkeznek az (1),(2),(3) tulajdonságokkal. Legyen ezek után

$$\varphi(\Omega_{I,J}) = \{\Omega_{I_0,J_0}, \Omega_{I_1,J_1}, \ldots, \Omega_{I_r,J_r}\}.$$

A φ és g függvények definíciójából következik, hogy rendelkeznek a kívánt tulajdonságokkal.

III. Faépítési stratégia

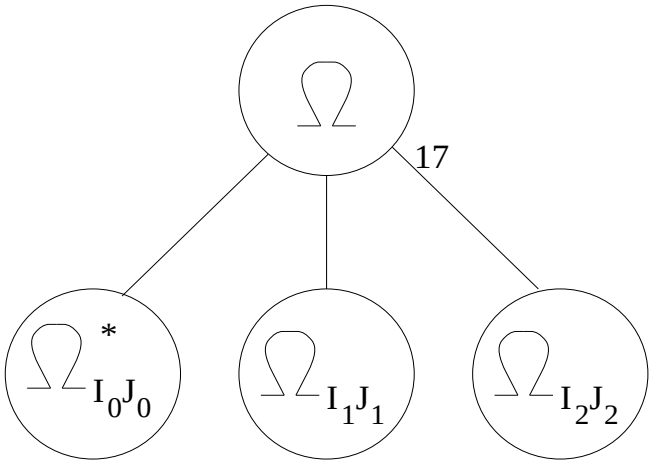
Azt a korábban már megismert stratégiát fogjuk alkalmazni, miszerint egy minimális korláttal rendelkező levelet választunk a faépítés során.

A $B\&B$ eljárás demonstrálására tekintsük az előzőekben vizsgált feladatot. Készítsünk egy lehetséges megoldást az $x_{12}^{(0)} = 1, x_{23}^{(0)} = 1, \ldots, x_{n1}^{(0)} = 1$ értékadásokkal. A megoldáshoz tartozó célfüggvényérték 38. Így tudunk kezdeti értéket adni $\mathbf{X}^*$ -nak és $\bar{z}$-nak.

A $g(\Omega)$ korlát meghatározásához a megfelelő hozzárendelési feladatot kell megoldanunk. Ezt már megtettük. Azt kaptuk, hogy az optimális megoldás három részkörút uniója, melyek az $(1,2),(2,3),(3,1)$, $(4,5),(5,6),(6,4)$, $(7,8),(8,7)$ éleket tartalmazzák, és az optimum értéke 17. Így $F_0 = \{\Omega\}$ és $g(\Omega) = 17$. Alkalmazzuk Ω-ra a φ szétválasztási függvényt:

$$\Omega_{I_0,J_0} = \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{78} = x_{87} = 1\},$$
$$\Omega_{I_1,J_1} = \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{78} = 0\},$$
$$\Omega_{I_2,J_2} = \{\mathbf{X} : \mathbf{X} \in \Omega \ \& \ x_{87} = 0 \ \& \ x_{78} = 1\}.$$

Mivel a $(7,8),(8,7)$ élekből álló részkörutat Ω_{I_0,J_0} minden eleme tartalmazza, ezért $\Omega_{I_0,J_0} \cap L = \emptyset$. Lássuk el az Ω_{I_0,J_0} halmazt $*$-gal. Ábrázolva az eddig létrejött $B\&B$-fát, a következő fát kapjuk:



10.3. ábra. Az eddig meghatározott B&B fa.

Ezek után a korlátok meghatározásával folytatódik az algoritmus. Definíció szerint $g(\Omega_{I_0,J_0}) = W$. A $g(\Omega_{I_1,J_1})$ korlát meghatározásához az $(A(\mathbf{C}), I_1, J_1)$ hozzárendelési feladatot kell megoldanunk. Mivel $\mathbf{C} \sim \mathbf{C}^{(0)}$, ezért szorítkozhatunk az $(A(\mathbf{C}^{(0)}), I_1, J_1)$ feladat megoldására. Ez azért praktikus, mert $\mathbf{C}^{(0)}$ -ban ki van jelölve egy n-elemű független 0-rendszer, amelyből $n-1$ darab független 0-t fel tudunk használni $(A(\mathbf{C}^{(0)}), I_1, J_1)$ megoldása során, mint induló független 0-rendszert. Ennek következtében a magyar módszer egyetlen láncképzéssel véget ér. Mivel $I_1 = \emptyset$ és $J_1 = \{(7,8)\}$, ezért $\mathbf{C}^{(0)}$-ban a $c^{(0)}_{78}$ együtthatót kell csak W-re változtatnunk. Az előálló új költségmátrix a megőrzött független 0 elemekkel, valamint a magyar módszer befejeztével kapott mátrix a következő:

$$\begin{pmatrix} W & 0^* & 9 & 8 & 6 & 5 & 4 & 3 \\ 3 & W & 0^* & 7 & 7 & 3 & 5 & 6 \\ 0^* & 9 & W & 8 & 5 & 9 & 0 & 8 \\ 8 & 8 & 9 & W & 0^* & 8 & 7 & 9 \\ 7 & 9 & 7 & 2 & W & 0^* & 8 & 7 \\ 8 & 6 & 3 & 0^* & 9 & W & 9 & 7 \\ 5 & 8 & 9 & 7 & 6 & 9 & W & W \\ 4 & 9 & 9 & 9 & 5 & 2 & 0^* & W \end{pmatrix} \quad \begin{pmatrix} W & 0 & 12 & 11 & 9 & 8 & 7 & 0^* \\ 3 & W & 0^* & 7 & 7 & 3 & 5 & 0 \\ 0^* & 6 & W & 8 & 5 & 9 & 0 & 2 \\ 8 & 5 & 9 & W & 0^* & 8 & 7 & 3 \\ 7 & 6 & 7 & 2 & W & 0^* & 8 & 1 \\ 8 & 3 & 3 & 0^* & 9 & W & 9 & 1 \\ 0 & 0^* & 4 & 2 & 1 & 4 & W & W \\ 4 & 6 & 9 & 9 & 5 & 2 & 0^* & W \end{pmatrix}$$

A tekintett $(A(\mathbf{C}^{(0)}), I_1, J_1)$ hozzárendelési feladat optimális megoldása: $x^{(1)}_{18} = x^{(1)}_{23} = x^{(1)}_{31} = x^{(1)}_{45} = x^{(1)}_{56} = x^{(1)}_{64} = x^{(1)}_{72} = x^{(1)}_{87} = 1$ és $x^{(1)}_{ij} = 0$ a többi indexekre.

Az optimum értéke $z(\mathbf{X}^{(1)}) = 28$, így $g(\Omega_{I_1,J_1}) = 28$. Az optimális megoldás nem körút, hanem az $(1,8),(8,7),(7,2),(2,3),(3,1)$ és a $(4,5)$, $(5,6),(6,4)$ éleket tartalmazó részkörutak uniója.

A $g(\Omega_{I_2,J_2})$ korlát meghatározásához az $(A(\mathbf{C}^{(0)}), I_2, J_2)$ feladatot fogjuk megoldani. Mivel $I_2 = \{(7,8)\}$ és $J_2 = \{(8,7)\}$, ezért $c_{87}^{(0)}$ és $c_{7t}^{(0)}$ $(t = 1,\ldots,7)$, $c_{s8}^{(0)}$ $(s = 1,\ldots,8; s \neq 7)$ együtthatókat W-re változtatjuk. Az előálló feladat költségmátrixa a megőrzött független 0-rendszerrel, valamint a magyar módszer végén kapott mátrix a következő:

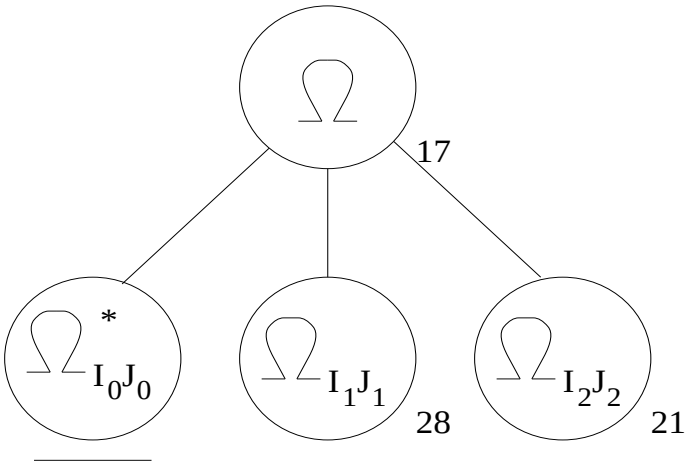
$$\begin{pmatrix} W & 0^* & 9 & 8 & 6 & 5 & 4 & W \\ 3 & W & 0^* & 7 & 7 & 3 & 5 & W \\ 0^* & 9 & W & 8 & 5 & 9 & 0 & W \\ 8 & 8 & 9 & W & 0^* & 8 & 7 & W \\ 7 & 9 & 7 & 2 & W & 0^* & 8 & W \\ 8 & 6 & 3 & 0^* & 9 & W & 9 & W \\ W & W & W & W & W & W & W & 0^* \\ 4 & 9 & 9 & 9 & 5 & 2 & W & W \end{pmatrix} \quad \begin{pmatrix} W & 0^* & 9 & 8 & 6 & 5 & 2 & W \\ 1 & W & 0^* & 7 & 7 & 3 & 3 & W \\ 0 & 11 & W & 10 & 7 & 11 & 0^* & W \\ 6 & 8 & 9 & W & 0^* & 8 & 5 & W \\ 5 & 9 & 7 & 2 & W & 0^* & 6 & W \\ 6 & 6 & 3 & 0^* & 9 & W & 7 & W \\ W & W & W & W & W & W & W & 0^* \\ 0^* & 7 & 7 & 7 & 3 & 0 & W & W \end{pmatrix}$$

A tekintett $(A(\mathbf{C}^{(0)}), I_2, J_2)$ feladat optimális megoldása:

$x_{12}^{(2)} = x_{23}^{(2)} = x_{37}^{(2)} = x_{45}^{(2)} = x_{56}^{(2)} = x_{64}^{(2)} = x_{78}^{(2)} = x_{81}^{(2)} = 1$ és $x_{ij}^{(2)} = 0$ a többi változóra.

Az optimum értéke $z(\mathbf{X}^{(2)}) = 21$, így $g(\Omega_{I_2,J_2}) = 21$. Az optimális megoldás nem körút, hanem az $(1,2),(2,3),(3,7),(7,8),(8,1)$, valamint a $(4,5),(5,6),(6,4)$ élekből álló részkörutak uniója.

Az iterációs lépés végén előálló $B\&B$ fa a következő:



10.4. ábra. Az iterációs lépés végén kapott B&B fa.

Az élő leveleket tartalmazó halmaz: $F_1 = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}$. A két osztály közül Ω_{I_2,J_2}-höz tartozik minimális korlát. $(A(\mathbf{C}^{(0)}), I_2, J_2)$ optimális megoldása nem körút, hanem két részkörútból áll. Ezek közül a $(4,5),(5,6),(6,4)$

éleket tartalmazó részkörút a minimális elemszámú. Alkalmazva Ω_{I_2,J_2}-re a φ szétválasztási függvényt, az alábbi osztályokhoz jutunk:

$$\Omega_{I_3,J_3} = \{\mathbf{X} : \mathbf{X} \in \Omega_{I_2,J_2} \ \& \ x_{45} = x_{56} = x_{64} = 1\},$$
$$\Omega_{I_4,J_4} = \{\mathbf{X} : \mathbf{X} \in \Omega_{I_2,J_2} \ \& \ x_{45} = 0\},$$
$$\Omega_{I_5,J_5} = \{\mathbf{X} : \mathbf{X} \in \Omega_{I_2,J_2} \ \& \ x_{56} = 0 \ \& \ x_{45} = 1\},$$
$$\Omega_{I_6,J_6} = \{\mathbf{X} : \mathbf{X} \in \Omega_{I_2,J_2} \ \& \ x_{64} = 0 \ \& \ x_{45} = x_{56} = 1\}.$$

Mivel $\Omega_{I_3,J_3} \cap L = \emptyset$, ezért $g(\Omega_{I_3,J_3}) = W$. A $g(\Omega_{I_4,J_4})$ korlát megadásához tekintsük az $(A(\mathbf{C}^{(0)}), I_4, J_4)$ hozzárendelési feladatot. Most $I_4 = \{(7,8)\}$ és $J_4 = \{(8,7),(4,5)\}$. Képezve $\mathbf{C}^{(0)}$-ból a tekintett feladatnak megfelelő költségmátrixot, majd végrehajtva a magyar módszert, az induló és befejező mátrixok a következők lesznek.

$$\begin{pmatrix} W & 0^* & 9 & 8 & 6 & 5 & 4 & W \\ 3 & W & 0^* & 7 & 7 & 3 & 5 & W \\ 0^* & 9 & W & 8 & 5 & 9 & 0 & W \\ 8 & 8 & 9 & W & W & 8 & 7 & W \\ 7 & 9 & 7 & 2 & W & 0^* & 8 & W \\ 8 & 6 & 3 & 0^* & 9 & W & 9 & W \\ W & W & W & W & W & W & W & 0^* \\ 4 & 9 & 9 & 9 & 5 & 2 & W & W \end{pmatrix} \quad \begin{pmatrix} W & 0^* & 9 & 8 & 3 & 5 & 4 & W \\ 3 & W & 0^* & 7 & 4 & 3 & 5 & W \\ 0^* & 9 & W & 8 & 2 & 9 & 0 & W \\ 1 & 1 & 2 & W & W & 1 & 0^* & W \\ 7 & 9 & 7 & 2 & W & 0^* & 8 & W \\ 8 & 6 & 3 & 0^* & 6 & W & 9 & W \\ W & W & W & W & W & W & W & 0^* \\ 2 & 7 & 7 & 7 & 0^* & 0 & W & W \end{pmatrix}$$

Az optimális megoldás: $x_{12}^{(4)} = x_{23}^{(4)} = x_{31}^{(4)} = x_{47}^{(4)} = x_{56}^{(4)} = x_{64}^{(4)} = x_{78}^{(4)} = x_{85}^{(4)} = 1$ és $x_{ij}^{(4)} = 0$ a többi indexekre. Az optimális megoldás nem körút. Az optimum értéke $z(\mathbf{X}^{(4)}) = 29$.

$g(\Omega_{I_5,J_5})$ meghatározásához tekintsük az $(A(\mathbf{C}^{(0)}), I_5, J_5)$ hozzárendelési feladatot. Ekkor $I_5 = \{(7,8),(4,5)\}$, $J_5 = \{(8,7),(5,6)\}$, és a fentieknek megfelelő mátrixok a következők:

$$\begin{pmatrix} W & 0^* & 9 & 8 & W & 5 & 4 & W \\ 3 & W & 0^* & 7 & W & 3 & 5 & W \\ 0^* & 9 & W & 8 & W & 9 & 0 & W \\ W & W & W & W & 0^* & W & W & W \\ 7 & 9 & 7 & 2 & W & W & 8 & W \\ 8 & 6 & 3 & 0^* & W & W & 9 & W \\ W & W & W & W & W & W & W & 0^* \\ 4 & 9 & 9 & 9 & W & 2 & W & W \end{pmatrix} \quad \begin{pmatrix} W & 0^* & 8 & 9 & W & 3 & 0 & W \\ 0 & W & 0^* & 9 & W & 2 & 2 & W \\ 0 & 13 & W & 13 & W & 11 & 0^* & W \\ W & W & W & W & 0^* & W & W & W \\ 0^* & 6 & 3 & 0 & W & W & 1 & W \\ 3 & 5 & 1 & 0^* & W & W & 4 & W \\ W & W & W & W & W & W & W & 0^* \\ 0 & 9 & 8 & 10 & W & 0^* & W & W \end{pmatrix}$$

Az optimális megoldás: $x_{12}^{(5)} = x_{23}^{(5)} = x_{37}^{(5)} = x_{45}^{(5)} = x_{51}^{(5)} = x_{64}^{(5)} = x_{78}^{(5)} = x_{86}^{(5)} = 1$ és $x_{ij}^{(5)} = 0$ a többi indexekre. Ez a megoldás körút. Az optimum értéke $z(\mathbf{X}^{(5)}) = 26$.

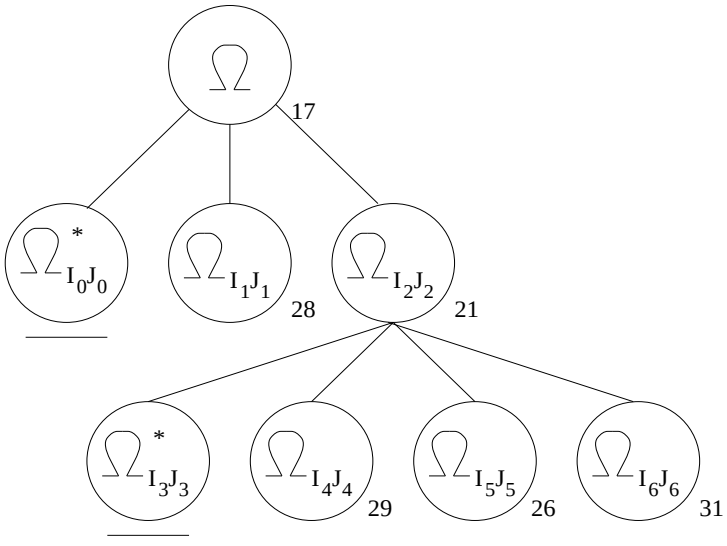
Végül határozzuk meg a $g(\Omega_{I_6,J_6})$ korlátot. Most $I_6 = \{(7,8),(4,5),(5,6)\}$ és $J_6 = \{(8,7),(6,4)\}$. Akkor az előzőeknek megfelelően az induló és befejező mátrixok a következők:

$$\begin{pmatrix} W & 0^* & 9 & 8 & W & W & 4 & W \\ 3 & W & 0^* & 7 & W & W & 5 & W \\ 0^* & 9 & W & 8 & W & W & 0 & W \\ W & W & W & W & 0^* & W & W & W \\ W & W & W & W & W & 0^* & W & W \\ 8 & 6 & 3 & W & W & W & 9 & W \\ W & W & W & W & W & W & W & 0^* \\ 4 & 9 & 9 & 9 & W & W & W & W \end{pmatrix} \quad \begin{pmatrix} W & 0^* & 9 & 1 & W & W & 2 & W \\ 1 & W & 0 & 0^* & W & W & 3 & W \\ 0 & 11 & W & 3 & W & W & 0^* & W \\ W & W & W & W & 0^* & W & W & W \\ W & W & W & W & W & 0^* & W & W \\ 3 & 3 & 0^* & W & W & W & 4 & W \\ W & W & W & W & W & W & W & 0^* \\ 0^* & 7 & 7 & 0 & W & W & W & W \end{pmatrix}$$

Így a tekintett feladat optimális megoldása:

$x_{12}^{(6)} = x_{24}^{(6)} = x_{37}^{(6)} = x_{45}^{(6)} = x_{56}^{(6)} = x_{63}^{(6)} = x_{78}^{(6)} = x_{81}^{(6)} = 1$ és $x_{ij}^{(6)} = 0$ a többi indexekre. A megoldás körút, az optimum értéke $z(\mathbf{X}^{(6)}) = 31$.

A korlátok meghatározása során két körutat is kaptunk, az $\mathbf{X}^{(5)}$ és $\mathbf{X}^{(6)}$ körutakat rendre 26 és 31 célfüggvényértékekkel. Így $\mathbf{X}^*$ és $\bar{z}$ értékeit aktualizálva, $\mathbf{X}^*$ új értéke $\mathbf{X}^{(5)}$ és $\bar{z}$ új értéke 26. Ekkor a megfelelő $B\&B$ fa a következő:



10.5. ábra. Az előállított B&B fa.

Mivel $\bar{z} = 26$ nem nagyobb egyik alsó korlátnál sem, ezért a fa valamennyi levele lezárásra kerül, azaz $F_2 = \emptyset$. Így a tekintett utazó ügynök probléma optimális megoldása $\mathbf{X}^{(5)}$, az optimum értéke pedig 26.

Az utazó ügynök probléma megoldására igen sok $B\&B$ eljárás kidolgozásra került. Ezek alapvetően a korlátozó függvényben, a szétválasztási függvényben, valamint a felbontandó levél kiválasztásának stratégiájában különböznek egymástól.

A hozzárendelési feladat felhasználását a korlát meghatározására számos szerző alkalmazta, többek között a [20], [30], [46], [128], [162], [165] dolgozatok tartalmaznak olyan eljárásokat, amelyekben a korlátozó függvény definiálása a tárgyaltakhoz hasonlóan történik. A szétválasztási függvényt illetően azonos technikát alkalmaztak a [20], [145], [165] munkákban közölt eljárásoknál. Ennek a módszernek egy olyan módosítása került bevezetésre a [30] dolgozatban, amelynél nem egy minimális elemszámú részkörutat választunk, hanem a részkörutak közül azt, amely minimális számú szabad változót fog eredményezni. (Ezzel csökken az aktuális $B\&B$-fában az adott szögponthoz tartozó leszármazottak száma.)

Más elven működő $B\&B$ eljárások is kidolgozásra kerültek, melyeket itt nem érintünk. A különböző technikák egy igen jó, összefoglaló tárgyalása megtalálható a [123] könyvben. Egy modern összefoglaló munka a [82] könyv.

11. Utazó ügynök heurisztikák

A közelítő eljárások létjogosultsága, hasznossága többféleképp indokolható. Az NP-nehéz problémák esetén nem ismeretesek polinomkorlátos műveletigényű megoldó eljárások. A rendelkezésre álló algoritmusok műveletigénye általában exponenciális függvénye a probléma méretének. Ennek következtében a nagyobb méretű feladatok megoldásának időigénye annyira nagy, hogy esetenként a megoldás nem realizálható. Így az NP-nehéz problémák esetében létjogosultsága van közelítő eljárásoknak. Ezek nem a tényleges optimális megoldást szolgáltatják, hanem egy lehetséges megoldást eredményeznek csak. Az így előállított lehetséges megoldással szemben alapvető elvárás, hogy "jó" legyen abban az értelemben, miszerint a hozzátartozó célfüggvényérték közel legyen az optimumértékhez. Az ilyen típusú eljárásokat szokásos *heurisztikus eljárásoknak* vagy egyszerűen *heurisztikáknak* nevezni.

A korábbiakban megismerkedtünk a $B\&B$ eljárással, amely igen sok NP-nehéz probléma esetében egy nagy műveletigényű algoritmus felépítését teszi lehetővé. Ezen eljárásokban a hatékonyság javítására felhasználhatók a rendelkezésre álló lehetséges megoldások. Minél jobb lehetséges megoldással rendelkezünk, általában annál gyorsabb, hatékonyabb lesz a $B\&B$ eljárás. Következésképp fontos az egyes problémacsoportokra olyan heurisztikák kidolgozása, amelyek gyorsak (polinomkorlátos műveletigényűek) és olyan lehetséges megoldást szolgáltatnak, melynek célfüggvényértéke közel van az optimumértékhez.

Az elmondottak kapcsán rögtön felvetődik a kérdés, hogy egy adott heurisztikát miként lehet minősíteni. Mikor mondhatjuk, hogy valamely heurisztika jó? Az elmúlt évtizedekben a heurisztikák vizsgálatára az alábbi három módszer alakult ki:

(1) legrosszabb esetek vizsgálata,

(2) valószínűségi analízis,

(3) empirikus analízis.

A következőkben rövid áttekintést adunk ezen módszerekről.

Legrosszabb esetek vizsgálata

Tekintsünk egy $\mathcal{C}$ minimalizálási problémaosztályt és jelöljön $\mathbf{A}$ egy, a tekintett problémák közelítő megoldását szolgáltató heurisztikát. A problémaosztály tetszőleges P problémájára jelölje $\mathbf{A}(P)$ a heurisztika által szolgáltatott lehetséges megoldáson felvett célfüggvényértéket és $\mathrm{OPT}(P)$ az optimumértéket. Ekkor az $\mathbf{A}$ algoritmust egy c konstansra c - *approximációs* algoritmusnak nevezzük, ha minden P probléma esetén $\mathbf{A}(P) \leq c \cdot \mathrm{OPT}(P)$. A legkisebb c értéket, amelyre az algoritmus c - approximációs az algoritmus *approximációs hányadosának* nevezzük. Az approximációs hányados egy további változata az *aszimptotikus approximációs hányados.* Az algoritmusra minden pozitív egész n-re definiáljuk az $R_n(\mathbf{A}) = \sup\{\mathbf{A}(P)/\mathrm{OPT}(P) : P \in \mathcal{C}, \mathrm{OPT}(P) \geq n\}$ értéket, és amennyiben létezik az $M_{\mathbf{A}} = \limsup R_n(\mathbf{A})$ érték, az adja meg a *heurisztika aszimptotikus approximációs hányadosát.* Az aszimptotikus approximációs hányados és az approximációs hányados közötti lényeges különbség az, hogy az aszimptotikus hányados azon inputokra, ahol az optimális költség kicsi nem követeli meg, hogy a heurisztika által adott megoldás az optimumhoz közeli érték legyen.

Az approximációs hányados illetve az aszimptotikus approximációs hányados nagysága a különböző problémaosztályokra és a különböző heurisztikákra más és más. Igen jó, 1 és 2 közé eső aszimptotikus approximációs hányadosok adódtak számos ládapakolási heurisztikára. Ezzel szemben az általános utazó ügynök problémára 1976-ban S. Sahni és T. Gonzalez [157] igazolták, hogy amennyiben létezik konstans approximációs hányados valamely polinomkorlátos TSP-heurisztikára, akkor P=NP.

Az említett negatív eredmény alapján akár el is vethetnénk a polinomkorlátos TSP heurisztikák gondolatát. Azonban más problémaosztályoknál nyert tapasztalatok alapján a legrosszabb esetek vizsgálatának eredménye és az algoritmus gyakorlatban történő viselkedése között lényeges eltérés lehet. Erre szemléletes példa a szimplex algoritmus. Az igen sok megoldásra került lineáris programozási feladat alapján az átlagos iterációs lépésszám közelítőleg $3n$, ahol n a feladat egyenleteinek számát jelöli. Másrészt ismert olyan lineáris programozási feladat (ld. [109]), amelyre az iterációs lépésszám 2^n. További ilyen jellegű észrevételek azt támasztják alá, hogy az approximációs hányados bár lényeges információt ad az algoritmusról, de ennek alapján nem ítélhető meg maradéktalanul az algoritmus jósága.

További érv a TSP heurisztikák vizsgálatához, hogy bizonyos speciális TSP feladatosztályokra léteznek viszonylag jó approximációs hányadossal rendelkező TSP heurisztikák.

Valószínűségi analízis

Ezekben a vizsgálatokban rögzített feladatméret mellett olyan P feladatot tekintenek, amelyben a paraméterek (együtthatók), mint független valószínűségi változók vannak megadva. Ekkor $\mathbf{A}(P)$ és $OPT(P)$, valamint hányadosuk is valószínűségi változó, melynek várható értékéből egy átlagos eltérésre lehet következtetni. Az így előállított mutatóval szemben kifogásolható, hogy a paramétereket megadó valószínűségi változók eloszlásaira tett feltételek nagyban befolyásolják azt, ugyanakkor a feltételek nem feltétlenül a problémaosztály jellemző tulajdonságait tükrözik. (Általában az egyenletes eloszlást teszik fel a feladatban szereplő valószínűségi változókra.)

Empirikus analízis

Az ilyen típusú vizsgálatokban konkrét problémamegoldásokból nyert eredmények alapján lehet bizonyos mutatókat képezni. A számítástechnika fejlődése, a gyors, nagyteljesítményű számítógépek megjelenése lehetővé tette azonos típusú nagyszámú probléma megoldását. Ezt kihasználva, rögzített számtartományból valamilyen (általában egyenletes) eloszlás mellett véletlenszerűen választva együtthatókat, majd az előállított konkrét problémáknak meghatározva az optimális, valamint a heurisztikus megoldását, képezhető a két célfüggvényérték hányadosa. Ezt elég sokszor ismételve, és átlagolva a hányadosokat, egyfajta empirikus mérőszámot kapunk. A vázolt vizsgálatban megkérdőjelezhető a választott eloszlás és számtartomány.

A fenti vázlatos áttekintésből kitűnik, hogy a heurisztikák jóságának vizsgálata igen bonyolult és csak részben megvalósítható feladat. Általában a heurisztikus eljárásokról az érdeklődők további meggondolásokat és részleteket találhatnak a [68] és [149] könyvekben. Mi a továbbiakban az egyszerűbb TSP heurisztikák közül fogunk néhányat ismertetni.

Az elmúlt időszakban számos TSP heurisztika került kidolgozásra és jelenleg is több kutató foglalkozik további eljárások felépítésével, illetve a meglévő eljárások vizsgálatával, javításával (ld. pl. [71], [101], [131], [152]). A különböző algoritmusok alapvetően három csoportba sorolhatók:

(1) körútépítő eljárások,

(2) körút javítását szolgáló algoritmusok,

(3) kombinált eljárások ((1) és (2) kombinációi).

A továbbiakban ismertetésre kerülő heurisztikák egy eljárás kivételével az (1) csoportba tartoznak. A tárgyalás egyszerűsítésének érdekében jelölje N az $\{1, \ldots, n\}$ halmazt.

Nearest addition

Előkészítő rész. Legyen $r = 1$, $I_r = \{1\}$, $E_r = \{(1,1)\}$. (Az 1 város lesz az induló részkörút.) Térjünk rá az iterációs eljárásrészre.

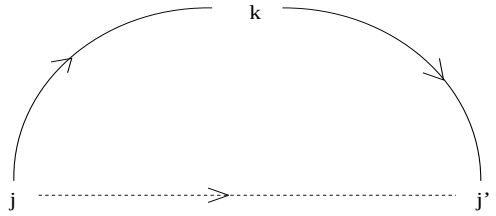
Iterációs rész (r. iteráció)

Ha $r = n$, akkor vége az eljárásnak, az E_r-beli élekből álló körút az eljárással szolgáltatott lehetséges megoldás. Ellenkező esetben határozzunk meg egy olyan $j \in I_r$, $k \in N \setminus I_r$ indexpárt, amelyre

$$c_{jk} = \min_{t \in N \setminus I_r} \{\min\{c_{st} : s \in I_r\}\} .$$

Legyen $I_{r+1} = I_r \cup \{k\}$. Mivel $j \in I_r$, ezért pontosan egy olyan $j' \in I_r$ index van, amelyre $(j, j') \in E_r$ ($r = 1$ esetén $j = j'$). Ekkor legyen $E_{r+1} = (E_r \setminus \{(j, j')\}) \cup \{(j, k), (k, j')\}$. Növeljük r értékét 1-gyel, és térjünk rá a következő iterációs lépésre.

Az eljárás úgy szemléltethető, hogy rendelkezésünkre áll egy részkörút ($r = 1$ esetén egyelemű részkörút). Ehhez kiválasztjuk a legközelebbi körúton kívüli várost. (Itt a legközelebbi város úgy értendő, hogy vesszük a körút pontjaiból a körúton kívüli pontokba vezető összes élet és ezen élek súlyainak a minimumát, ez a c_{jk} mennyiség.) Az így kiválasztott várossal bővítjük a körutat úgy, hogy a hozzá legközelebb eső, a körútban szereplő j városból ebbe a városba látogatunk, majd innen a j város j' leszármazottjába. Az alábbi ábra mutatja a beszúrási technikát. Az aktuális részkörútból töröljük a (j, j') élet és bővítjük a $(j, k), (k, j')$ élekkel.



11.1. ábra. Beszúrási technika.

A számításokat nagyban megkönnyíti egy olyan távolságvektor használata, amely minden iterációs lépésben megadja az aktuális részkörúton kívüli városoknak a részkörúttól való távolságát. Az első iterációs lépésben ez a $\mathbf{C}$ mátrix első sorvektora az $(1,1)$ indexű elem kivételével. Egyszerűen belátható, hogy amennyiben $\mathbf{v}_r$ az r-edik iterációs lépésben a távolságvektor, és ebben a lépésben a k várossal bővítjük a részkörutat, akkor $\mathbf{v}_{r+1}$-et megkapjuk úgy, hogy a körúton kívüli városokra rendre képezzük a $\mathbf{v}_r$ vektor megfelelő komponensének és a $\mathbf{C}$ mátrix k-adik sorában lévő megfelelő célfüggvényegyütthatónak a minimumát.

Az eljárás demonstrálására alkalmazzuk a fenti heurisztikát az előzőekben vizsgált 8-városos feladatra. Ekkor $\mathbf{v}_1 = (-, 2, 11, 10, 8, 7, 6, 5)$. Így az aktuális részkörúthoz (jelenleg az 1 várost tartalmazó részkörút) legközelebb levő város a 2-vel jelzett. Bővítve 2-vel a részkörutat, az $(1,2),(2,1)$ élekből álló új részkörúthoz jutunk. A távolságvektort a következőképpen tudjuk aktualizálni. A $\mathbf{v}_1$ vektor j-edik komponensének és c_{2j}-nek kell képezni a minimumát, ez lesz a $\mathbf{v}_2$ vektor j-edik komponense, ahol $j = 3, \ldots, 8$. A teljes eljárás végrehajtását az alábbi táblázatban foglaltuk össze, ahol az aktuális részkörutat ciklikus permutációként adjuk meg.

r	$\mathbf{v}_r$	(j,k)	részkörút
1	$(-, 2^*, 11, 10, 8, 7, 6, 5)$	$(1,2)$	$(1,2)$
2	$(-, -, 1^*, 8, 8, 4, 6, 5)$	$(2,3)$	$(1,2,3)$
3	$(-, -, -, 8, 8, 4, 3^*, 5)$	$(3,7)$	$(1,2,3,7)$
4	$(-, -, -, 8, 8, 4, -, 3^*)$	$(7,8)$	$(1,2,3,7,8)$
5	$(-, -, -, 8, 6, 3^*, -, -)$	$(8,6)$	$(1,2,3,7,8,6)$
6	$(-, -, -, 2^*, 6, -, -, -)$	$(6,4)$	$(1,2,3,7,8,6,4)$
7	$(-, -, -, -, 1^*, -, -, -)$	$(4,5)$	$(1,2,3,7,8,6,4,5)$

A tekintett feladatra az eljárás most pontosan egy optimális megoldást eredményezett. Ez egy kicsit megtévesztő. A vizsgált heurisztika nem ilyen jó. Az előzőekből tudjuk, hogy az általános TSP-re a $P \neq NP$ feltétel mellett nem létezik konstans approximációs hányados polinomkorlátos heurisztikák esetében. A tárgyalt eljárásról egyszerűen belátható, hogy polinomkorlátos, mégpedig $O(n^2)$ műveletigénnyel, így általános esetben nem várható hozzá konstans approximációs hányados.

Nearest insertion

Az eljárás ugyanúgy épül fel, mint az előző. Az eltérés annyi, hogy nem a kiválasztott éllel bővítjük a részkörutat, hanem a kiválasztott, k-val jelölt városra meghatározzuk k lehető legjobb (legkisebb költséggel járó) beszúrását a részkörútba, és ezt hajtjuk végre.

Előkészítő rész. Legyen $r = 1$, $I_r = \{1\}$, $E_r = \{(1,1)\}$. Térjünk rá az iterációs eljárásrészre.

Iterációs rész (r. iteráció)

Ha $r = n$, akkor vége az eljárásnak, az E_r-beli élekből álló körút a heurisztikával előállított lehetséges megoldás. Ellenkező esetben határozzunk meg egy olyan $j \in I_r$, $k \in N \setminus I_r$ indexpárt, amelyre

$$c_{jk} = \min_{t \in N \setminus I_r} \{\min\{c_{st} : s \in I_r\}\} .$$

Legyen $I_{r+1} = I_r \cup \{k\}$, majd válasszunk egy olyan $(u,v) \in E_r$ élet, amelyre

$$\delta_{uv} = c_{uk} + c_{kv} - c_{uv} = \min\{c_{sk} + c_{kt} - c_{st} : (s,t) \in E_r\}.$$

Legyen $E_{r+1} = (E_r \setminus \{(u,v)\}) \cup \{(u,k),(k,v)\}$. Növeljük r értékét 1-gyel, és folytassuk az eljárást a következő iterációs lépéssel.

Ezen eljárásnál is használható az előzőekben bevezetett távolságvektor. A heurisztika bemutatására ismételten a 8-városos feladatot fogjuk használni. A végrehajtás egyes lépéseit a következő táblázat tartalmazza.

r	$\mathbf{v}_r$	k	(u,v)	δ_{uv}	részkörút
1	$(-, 2^*, 11, 10, 8, 7, 6, 5)$	2	$(1,1)$	-	$(1,2)$
2	$(-, -, 1^*, 8, 8, 4, 6, 5)$	3	$(2,1)$	0	$(1,2,3)$
3	$(-, -, -, 8, 8, 4, 3^*, 5)$	7	$(3,1)$	8	$(1,2,3,7)$
4	$(-, -, -, 8, 8, 4, -, 3^*)$	8	$(7,1)$	0	$(1,2,3,7,8)$
5	$(-, -, -, 8, 6, 3^*, -, -)$	6	$(2,3)$	8	$(1,2,6,3,7,8)$
6	$(-, -, -, 2^*, 6, -, -, -)$	4	$(6,3)$	7	$(1,2,6,4,3,7,8)$
7	$(-, -, -, -, 1^*, -, -, -)$	5	$(4,3)$	0	$(1,2,6,4,5,3,7,8)$

A kapott körúthoz tartozó célfüggvényérték 31, az optimum értéke 26, így a hányados 1.19. Az eljárással kapcsolatban megemlítjük, hogy műveletigénye $O(n^2)$, továbbá olyan speciális TSP feladatokra, melyek költségmátrixá-

ra teljesül a háromszögegyenlőtlenség és a költségmátrix szimmetrikus igazolást nyert (ld. [155]), hogy ebben az esetben az approximációs hányados 2.

Farthest insertion

Az eljárás analóg az előzőhöz. Az eltérés csupán annyi, hogy az aktuális részkörúttól legtávolabb levő várost választjuk ki, és ezt szúrjuk be a lehető legkisebb költséggel a részkörútba.

Itt is használható a távolságvektor. A heurisztikát ismét a 8-városos problémára alkalmazzuk, és a végrehajtás lépéseinek eredményét az alábbi táblázat tartalmazza.

r	$\mathbf{v}_r$	k	(u,v)	δ_{uv}	részkörút
1	$(-,2,11^*,10,8,7,6,5)$	3	$(1,1)$	-	$(1,3)$
2	$(-,2,-,10^*,8,7,3,5)$	4	$(1,3)$	9	$(1,4,3)$
3	$(-,2,-,-,1,7^*,3,5)$	6	$(1,4)$	-1	$(1,6,4,3)$
4	$(-,2,-,-,1,-,3,5^*)$	8	$(1,6)$	1	$(1,8,6,4,3)$
5	$(-,2^*,-,-,1,-,1,-)$	2	$(4,3)$	0	$(1,8,6,4,2,3)$
6	$(-,-,-,-,1^*,-,1,-)$	5	$(4,2)$	3	$(1,8,6,4,5,2,3)$
7	$(-,-,-,-,-,-,1,-)$	7	$(1,8)$	4	$(1,7,8,6,4,5,2,3)$

A kapott körúthoz tartozó célfüggvényérték 32.

Cheapest insertion

Hasonló az előzőekhez. A lényeges eltérés a részkörút bővítésére szolgáló város kiválasztásában van. Minden, az aktuális részkörúton kívüli k ($\in N \setminus I_r$) városra kiszámítjuk, hogy mennyi az aktuális részkörútba történő beszúrásának minimális költsége figyelembe véve az összes lehetséges beszúrásokat. Így minden $k \in N \setminus I_r$ városra adódik egy beszúrási költség. Ezek közül választunk egy minimálisat, és az ennek megfelelő város (minimális költségű) beszúrásával bővítjük a körutat.

A heurisztika végrehajtását a vizsgált példán szemléltetjük.

r	k	(u,v)	δ_{uv}	részkörút
1	2	$(1,1)$	-	$(1,2)$
2	3	$(2,1)$	0	$(1,2,3)$
3	6	$(2,3)$	8	$(1,2,6,3)$
4	5	$(2,6)$	6	$(1,2,5,6,3)$
5	4	$(2,5)$	1	$(1,2,4,5,6,3)$
6	7	$(2,4)$	8	$(1,2,7,4,5,6,3)$
7	8	$(2,7)$	2	$(1,2,8,7,4,5,6,3)$

A kapott körúthoz tartozó célfüggvényérték 33. Az eljárás műveletigénye $O(n^3)$. A [63] dolgozatban igazolást nyert, hogy a már említett speciális TSP-k esetében (szimmetrikus költségmátrixúak, amelyekre teljesül a háromszögegyenlőtlenség) az eljárás approximációs hányadosa 2.

Tekintettel arra, hogy az ismertetett eljárások mindegyike igen gyors, továbbá az előállított körút függ a kiinduló várostól (mi ezt rendre 1-nek választottuk), lehetséges ugyanazt a heurisztikát többször is végrehajtani más és más kiindulási városokkal, majd az előálló körutak közül venni a legjobbat. Ilyenkor az illető eljárás *all cities változatáról* beszélünk.

A tárgyalt heurisztikákat illetően egymástól függetlenül A. M. Adrabinski és M. Syslo [1], valamint B. Golden és társai [76] végeztek empirikus vizsgálatokat. A kapott eredmények azt mutatják, hogy a farthest insertion eljárás jobb értékeket szolgáltat általában, mint a másik három algoritmus. A következő táblázatban szereplő számsorokat a [76] munkából gyűjtöttük ki. Az adatok 5 darab 100×100-as euklideszi TSP-re vonatkoznak. (A városok egy euklideszi tér valamely síkjában vannak és távolságuk a térben a pontok távolsága.) A táblázatban azt adjuk meg, hogy a közelítő megoldás célfüggvényértéke hány százaléka az optimum értékének.

Nearest insertion (all cities)	118.69	117.81	122.96	114.44	120.33
Farthest insertion (all cities)	105.14	106.97	103.17	101.99	107.42
Cheapest insertion (all cities)	112.07	111.67	120.83	112.97	112.16

A következő eredményeket Adrabinski és Syslo [1] munkájából gyűjtöttük ki. (A feladatok méretei rendre $n = 20, 27, 42, 57, 120$).

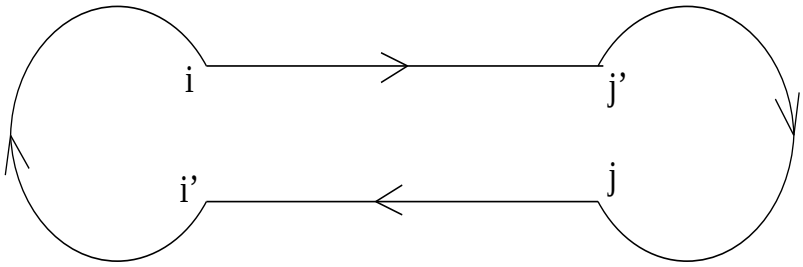
Nearest addition (one city)	186.58	113.88	111.02	114.76	121.78
Nearest insertion (one city)	138.21	106.89	110.87	109.25	117.27
Farthest insertion (one city)	128.45	100.35	101.43	101.28	103.06

Patching eljárások

A címben szereplő heurisztikák a hozzárendelési feladat és az utazó ügynök problémák kapcsolatára épülnek, és a következő észrevételen alapulnak. Oldjuk meg a megfelelő hozzárendelési feladatot. Ha az optimális megoldás körút, akkor megkaptuk a TSP optimális megoldását. Ellenkező esetben az optimális hozzárendelésnek megfelelő gráf diszjunkt részkörutak egyesítése. Ezen részkörutak számáról bizonyítást nyert, hogy amennyiben a költségmátrix elemeit egyenletes eloszlás alapján generáljuk, akkor a részkörutak számának várható értéke $\log(n)$. Így a részkörutak kis költséggel járó összekapcsolása, összefűzése egy jó heurisztikus megoldást eredményezhet. Attól függően, hogy miként kapcsoljuk össze a részkörutakat, különböző algoritmusok építhetők fel. A továbbiakban két ilyen típusú eljárást fogunk ismertetni.

2-patching eljárás

Tekintsünk két részkörutat. Jelölje I, J az egyes részkörutakban szereplő városok halmazát, továbbá tetszőleges k városra jelölje k' a k leszármazottját a k-t tartalmazó részkörútban. Legyen $i \in I$ és $j \in J$. Akkor törölve a részkörutakból az (i, i'), (j, j') éleket, és felvéve az (i, j'), (j, i') éleket, olyan részkörutat kapunk, amely az $I \cup J$-beli városokat tartalmazza. Az alábbi ábra mutatja az összekapcsolást.



11.2. ábra. A két részkörút összevonása.

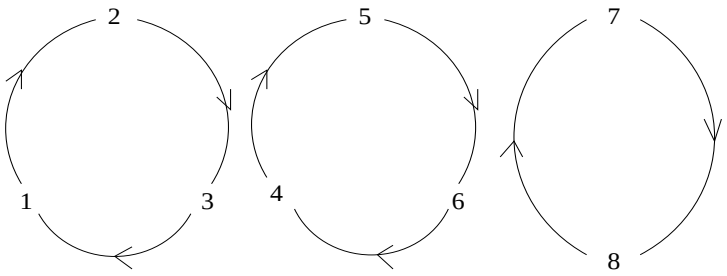
A két kör összefűzésével keletkező költséget egyszerűen megkaphatjuk, ez $c_{ij'} + c_{ji'} - c_{ii'} - c_{jj'}$. Véve ezeket a költségeket az összes $i \in I$ és $j \in J$ indexpárra, majd képezve ezek minimumát, megkapjuk a két részkör ilyen típusú "összekapcsolására" vonatkozó minimális költséget. Ezt *2-patching költségnek* és a részkörutak megfelelő összekapcsolását *2-patching műveletnek* nevezzük. Ezek után felépíthető az alábbi eljárás, amelyet R. M. Karp publikált 1979-ben a [106] dolgozatban.

Eljárás ([106])

Előkészítő rész. Oldjuk meg az $A(\mathbf{C})$ hozzárendelési feladatot. Ha $A(\mathbf{C})$ optimális megoldása körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá az iterációs eljárásrészre.

Iterációs rész. Válasszunk ki két legkisebb városszámú részkörutat és kapcsoljuk össze őket egy alkalmas 2-patching művelettel. Ha az előálló új megoldás (hozzárendelés) körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá a következő iterációra.

Az eljárás demonstrálására tekintsük ismét az előző példát. A hozzárendelési feladat optimális megoldása a következő három részkörútból áll.

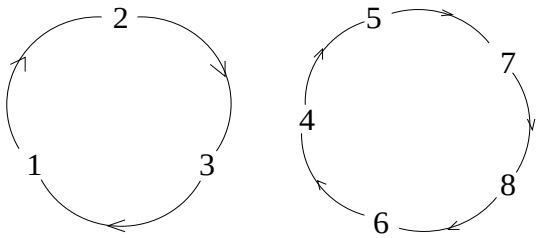


11.3. ábra. A kapott három részkörút.

Válasszuk ki a második és harmadik részkörutat. A kiválasztott részkörutakra a következő táblázatban adjuk meg a 2-patching költségeket. (A költségek számításánál az $\mathbf{A}^{(0)}$ mátrixot használtuk.)

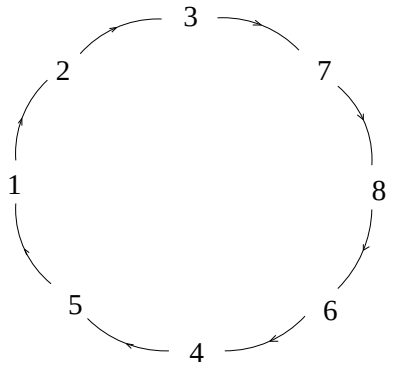
i	j	költség
4	7	$c^{(0)}_{48} + c^{(0)}_{75} - c^{(0)}_{45} - c^{(0)}_{78} = 9 + 6 = 15$
4	8	$c^{(0)}_{47} + c^{(0)}_{85} - c^{(0)}_{45} - c^{(0)}_{87} = 7 + 5 = 12$
5	7	$c^{(0)}_{58} + c^{(0)}_{76} - c^{(0)}_{56} - c^{(0)}_{78} = 7 + 9 = 16$
5	8	$c^{(0)}_{57} + c^{(0)}_{86} - c^{(0)}_{56} - c^{(0)}_{87} = 8 + 2 = 10^*$
6	7	$c^{(0)}_{68} + c^{(0)}_{74} - c^{(0)}_{64} - c^{(0)}_{78} = 7 + 7 = 14$
6	8	$c^{(0)}_{67} + c^{(0)}_{84} - c^{(0)}_{64} - c^{(0)}_{87} = 9 + 9 = 18$

A 2-patching költség 10. A megfelelő 2-patching művelet során az $(5, 7)$ és $(8, 6)$ éleket kell felvenni és az $(5, 6)$, $(8, 7)$ éleket kell törölni. Az új megoldás a 11.4. ábrán megadott két részkörútból áll:



11.4. ábra. Részkörutak az összevonás után.

Kiszámítva a fenti két körre a 2-patching költséget, az $i = 3$, $j = 5$ esetben kapjuk a minimumot. Törölve a $(3, 1)$, $(5, 7)$ éleket, valamint felvéve a $(3, 7)$, $(5, 1)$ éleket, az alábbi körutat kapjuk, amely éppen egy optimális megoldás.



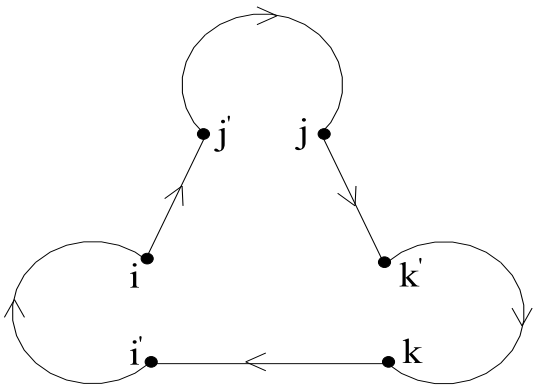
11.5. ábra. A 2-patching eljárással nyert optimális megoldás.

Az ismertetett 2-patching eljárásra J. M. Steele és R. M. Karp végzett valószínűségi analízist (ld. [123]). Igazolták, hogy a [0,1] intervallumon egyenletes eloszlású független valószínűségi változók esetén a közelítő érték és az optimumérték hányadosának várható értéke $1 + o(n^{-\frac{1}{2}})$.

3-patching eljárás

Kettőnél több kör összekapcsolása hatékonyabb eljárást eredményezhet. Ezen az észrevételen alapul a [13] dolgozatban közölt *3-patching eljárás*, melynek megadásához szükségesek bizonyos előkészületek.

Tekintsünk három részkörutat. Jelölje rendre I, J, K a részkörutakban szereplő városok halmazát. Legyen $i \in I$, $j \in J$, $k \in K$, továbbá jelölje i', j', k' rendre az i, j, k leszármazottait az I, J, K halmazokhoz tartozó részkörutakban. Akkor törölve az (i, i'), (j, j'), (k, k') éleket, és felvéve az (i, j'), (j, k'), (k, i') éleket, a három részkörutat egyetlen részkörútba (körútba) tudjuk egyesíteni. A 11.6. ábra szemlélteti a három részkörút egyesítését.



11.6. ábra. Három részkörút összekapcsolása.

A három részkörút tekintett összekapcsolásának költsége:

$$c_{ij'} + c_{jk'} + c_{ki'} - c_{ii'} - c_{jj'} - c_{kk'} \ .$$

Rendre meghatározva ezeket a költségeket minden $i \in I$, $j \in J$, $k \in K$ indexre, megkapjuk a három részkörút ilyen típusú összekapcsolására vonatkozó minimális költséget. Ezt a költséget *3-patching költségnek*, és a részkörutak ennek megfelelő összekapcsolását *3-patching műveletnek* nevezzük.

Ezek után felépíthető a következő heurisztikus eljárás.

Eljárás

Előkészítő rész. Oldjuk meg az $A(\mathbf{C})$ hozzárendelési feladatot. Ha $A(\mathbf{C})$ optimális megoldása körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá az iterációs eljárásrészre.

Iterációs rész

- *1. lépés.* Jelölje m az aktuális hozzárendelés részkörútjainak a számát. Ha $m \leq 9$, akkor a 2. lépés következik. Ellenkező esetben rendezzük a részkörutakat elemszámuk tágabb értelemben vett növekvő sorrendjébe. Jelölje a rendezett sorozatot $U_1, \ldots, U_m$. Számítsuk ki a d_{rs} 2-patching költségeket az U_r, U_{m-l+s} részkörutakra minden $1 \leq r \leq l$, $1 \leq s \leq l$ indexpárra, ahol $l = [m/2]$. Oldjuk meg a (d_{ij}) költségmátrixú $l \times l$-es hozzárendelési feladatot, és jelölje az optimális hozzárendelést φ. Hajtsunk végre egy $d_{r\varphi(r)}$ költségű 2-patching műveletet az $U_r, U_{m-l+\varphi(r)}$ részkörutakon minden $1 \leq r \leq l$ indexre. Az így képezett részkörutak által meghatározott hozzárendelést tekintve aktuális hozzárendelésnek, folytassuk az eljárást az 1. lépés ismétlésével.

- *2. lépés.* Ha az aktuális hozzárendelés körút, akkor vége az eljárásnak. Ellenkező esetben, ha $m < 3$, akkor hajtsunk végre a két részkörúton egy 2-patching műveletet, és vége az eljárásnak. $m \geq 3$ esetén határozzunk meg a részkörutak közül három olyan részkörutat, amelyek 3-patching költsége minimális. Hajtsunk végre a kiválasztott három részkörúton egy, a minimális költségnek megfelelő 3-patching műveletet. Az új hozzárendelést tekintve aktuális hozzárendelésnek, és részkörútjainak számát az aktuális m-nek, folytassuk az eljárást a 2. lépés ismétlésével.

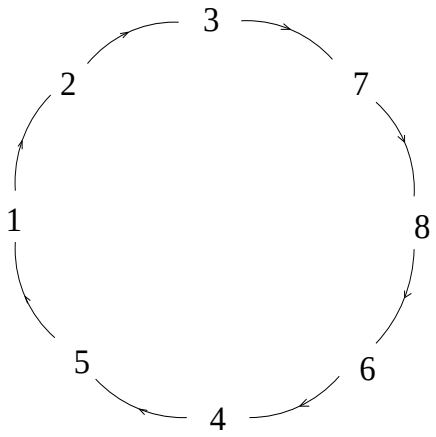
A fenti eljárásban $m > 9$ esetén a kis köröket párosítjuk a nagy körökkel, és olyan párosításra törekszünk, hogy az összepárosított köröket összefűzve, a költségek növekedése ne legyen jelentős. A 9 konstans használatát az indokolja, hogy fixpont nélküli permutációk esetében a részkörök számának várható értéke közelítőleg $\log(n)$, ahol n a városok számát jelöli. Végül az eljárás hatékony végrehajtásához bevezethető egy 3-dimenziós tömb, amely

a részkörutakra a 3-patching költségeket tartalmazza. Az első ilyen tömb alapján a további tömbök már viszonylag kis műveletigénnyel kiszámíthatók.

Az eljárás szemléltetésére tekintsük ismét a korábbiakban vizsgált 8-városos feladatot. A megfelelő hozzárendelési feladat optimális megoldása az $(1,2),(2,3),(3,1)$, a $(4,5),(5,6),(6,4)$ és a $(7,8),(8,7)$ éleket tartalmazó részkörutakból áll. A $c_{ij'}+c_{jk'}+c_{ki'}-c_{ii'}-c_{jj'}-c_{kk'}$ költségeket Δ_{ijk}-val jelölve, a konkrét értékeket a következő táblázatban adtuk meg:

i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}
1	4	7	23	2	5	8	20	1	7	4	17	2	8	4	19
1	4	8	22	2	6	7	23	1	7	5	21	2	8	5	14
1	5	7	20	2	6	8	25	1	7	6	16	2	8	6	17
1	5	8	22	3	4	7	19	1	8	4	17	3	7	4	22
1	6	7	23	3	4	8	16	1	8	5	15	3	7	5	24
1	6	8	26	3	5	7	21	1	8	6	19	3	7	6	23
2	4	7	25	3	5	8	21	2	7	4	21	3	8	4	13
2	4	8	23	3	6	7	20	2	7	5	22	3	8	5	9^*
2	5	7	19	3	6	8	21	2	7	6	16	3	8	6	17

Az $i=3$, $j=8$ és $k=5$ indexekre kapjuk a legkisebb értéket. Végrehajtva a megfelelő 3-patching műveletet, az alábbi körút adódik, amely speciálisan optimális megoldás is.



11.7. ábra. A 3-patching eljárással nyert optimális megoldás.

Az ismertetett eljárásokkal kapcsolatosan a [13] munkában egy empirikus analízis található, amelyben ezen eljárások nyertek összehasonlítást. A vizsgálatok során rendre minden tekintett méret mellett 100-100 probléma került generálásra. A célfüggvényegyütthatók a $0, 1, \ldots, 100$ egészek közül egyenletes eloszlás mellett lettek generálva. A számítógépes vizsgálat eredményét a következő táblázatban foglaltuk össze. Négy városszámra, az $n = 100$, $n = 150$, $n = 200$ és $n = 250$ városszámokra történt a 100-100 utazó ügynök probléma generálása. A táblázat baloldala tartalmazza a vizsgált eljárásokat, és a megfelelő sor adja meg az illető eljáráshoz tartozó értékeket. Minden városméretre az első oszlop adja meg az érintett eljárással meghatározott célfüggvényértékek valamint az optimumértékek hányadosainak átlagát, a második oszlop tartalmazza az érintett eljárás futási idejeinek átlagát, végül a harmadik oszlop azt mutatja, hogy az érintett eljárás hány esetben szolgáltatta a legjobb célfüggvényértéket. Ez a B&B eljárásnál üres maradt, mivel ez mindig az optimális megoldást szolgáltatja.

	$n = 100$			$n = 150$			$n = 200$			$n = 250$		
	average ratio	average sec.	best value	average ratio	average sec.	best value	average ratio	average sec.	best value	average ratio	average sec.	best value
B&B	1.000	40.78	-	1.000	87.69	-	1.000	194.1	-	1.000	320.9	-
3-patching k=5	1.054	19.53	88	1.056	58.55	81	1.052	190.2	82	1.059	370.8	80
3-patching k=3	1.061	11.60	70	1.063	34.93	67	1.061	122.0	54	1.078	218.6	48
3-patching k=1	1.069	3.84	55	1.096	11.90	39	1.094	39.8	25	1.134	72.6	24
2-patching k=5	1.090	11.14	33	1.082	29.70	38	1.069	88.4	40	1.101	158.3	37
2-patching k=3	1.092	6.69	28	1.097	17.92	26	1.085	53.1	27	1.119	94.8	23
2-patching k=1	1.108	2.21	21	1.127	6.04	15	1.127	17.7	13	1.177	31.5	12
cheapest insertion	4.654	7.77	0	6.794	37.48	0	9.934	89.7	0	18.11	175.4	0
nearest insertion	4.392	9.19	0	7.047	43.66	0	11.39	104.6	0	18.60	205.1	0
farthest insertion	4.534	8.76	0	7.110	43.71	0	11.39	104.6	0	18.60	205.3	0
nearest addition	18.43	7.09	0	33.84	32.72	0	57.51	77.0	0	98.43	149.7	0

A táblázatban megadott eredmények értelmezéséhez szükséges két további információ. A tekintett cheapest insertion, nearest insertion, farthest insertion, nearest addition heurisztikák mindegyike "all cities" változat, azaz a

generált feladatra n-szer hajtottuk végre az illető heurisztikus eljárást mindig más és más városból indulva, majd a kapott megoldások közül vettük a legjobbat.

Mivel minden patching eljárás a generált költségmárixhoz tartozó hozzárendelési feladat optimális megoldásából indul ki, ezért különböző optimális megoldásokból kiindulva, a patching eljárások a $TSP(\mathbf{C})$ probléma különböző lehetséges megoldásait szolgáltathatják, és ezek közül választhatjuk a legjobbat. Amikor hozzárendelési feladatot oldunk meg magyar módszerrel, akkor az optimális megoldás függhet az induló független 0-rendszertől, amit oszlopfolytonosan határozunk meg. Ha az oszlopokat nem indexeik növekvő sorrendjében, hanem random módon választjuk ki, akkor különböző induló független 0-rendszereket, és így a hozzárendelési feladat különböző optimális megoldásait kaphatjuk. A patching eljárásoknál a hozzárendelési feladat megoldása során az induló független 0-rendszer meghatározásához az oszlopokat random módon választottuk ki, az ebből nyert optimális megoldást használtuk a további eljárásban. Mindezt k-szor hajtottuk végre, és az előálló legfeljebb k lehetséges megoldás közül vettük a legjobbat. A k paramétert a táblázat első oszlopában az eljárás neve alatt tüntettük fel.

A fenti táblázat szépen mutatja, a következő tendenciákat. Valamennyi patching eljárás egészen jó szuboptimális megoldásokat szolgáltat, és ez a "jóság" nem változik szignifikánsan a feladatok méretének növekedésével. Ezzel szemben a különböző beszúrási eljárásokra azt tapasztalhatjuk, hogy az általuk meghatározott lehetséges megoldások célfüggvényértéke többszöröse az optimumértéknek és ez szignifikánsan romlik a feladatok méreteinek növekedésével.

Az eddig bemutatott heurisztikus eljárások mindegyike az általános esetre (a költségmátrixról nem tételezünk fel semmit) lett kifejlesztve. Ettől függetlenül ezek az eljárások alkalmazhatók olyan TSP feladatosztályokra, ahol a költségmátrixokra különböző megszorításokat teszünk. A bemutatott eljárásokhoz fűzött megjegyzések azt mutatják, hogy bizonyos megszorítások mellett a heurisztikák viselkedéséről többet tudunk állítani.

A továbbiakban speciális TSP feladatosztályokat vizsgálunk és ezekre kifejlesztett heurisztikus eljárásokat mutatunk be. Elsőként a szimmetrikus esetet vizsgáljuk ($c_{ij} = c_{ji}$, $i = 1, \ldots, n; j = 1, \ldots, n$), ami megfelel az irányítatlan gráfok esetének, és erre adunk meg egy körútjavító heurisztikus eljárást.

Az ismertetésre kerülő eljárás azon az észrevételen alapul, hogy amennyiben adott egy körút, úgy abból törölve két nem szomszédos élet, a körút két diszjunkt útra esik szét. Ezek után létezik két olyan egyértelműen

meghatározott él, hogy ezekkel bővítve a két útból álló gráfot, az eredmény egy másik körút lesz. (Például, ha az $(1,2),\dots(4,5),(1,5)$ élekből álló körútból töröljük a $(2,3)$ és $(4,5)$ éleket, akkor az előálló két útból csak a $(2,4)$ és $(3,5)$ élek felvételével készíthető körút.) A továbbiak egyszerűsítésének érdekében nevezzünk az $\bar{\mathbf{X}}$ *körút szomszédjának* minden olyan körutat, amely előáll $\bar{\mathbf{X}}$-ből két él törlésével, és két új él felvételével. Egyszerűen belátható, hogy $\bar{\mathbf{X}}^{n\times n}$ szomszédjainak a száma $n(n-3)/2$, ha önmagát nem számítjuk szomszédnak.

2-optimális eljárás

Előkészítő rész. Határozzuk meg valamilyen eljárással a feladat egy $\bar{\mathbf{X}}$ körútját. Legyen $\mathbf{X}^{(0)}=\bar{\mathbf{X}}$, $r=0$, és térjünk rá az iterációs részre.

Iterációs rész (r. iteráció)

- *1. lépés.* Határozzuk meg $\mathbf{X}^{(r)}$ összes szomszédját. Ha $\mathbf{X}^{(r)}$ minden $\bar{\mathbf{X}}$ szomszédjára $z(\mathbf{X}^{(r)})\le z(\bar{\mathbf{X}})$ teljesül, akkor vége az eljárásnak, $\mathbf{X}^{(r)}$ az eljárással előállított körút. Ellenkező esetben a 2. lépés következik.
- *2. lépés.* Jelöljön $\bar{\mathbf{X}}$ $\mathbf{X}^{(r)}$ szomszédjai közül egy olyan körutat, amelyen a z függvény a szomszédokra vonatkozóan minimális értéket vesz fel. Legyen $\mathbf{X}^{(r+1)}=\bar{\mathbf{X}}$, növeljük r értékét eggyel, és térjünk rá a következő iterációs lépésre.

Az eljárás bemutatására tekintsük a következő példát.

11.1. példa. Tekintsük azt az 5 városos utazó ügynök feladatot, amelynek költségmátrixa az alábbi $\mathbf{C}$ mátrix és legyen az induló körutunk az a körút, amely az

$$\mathbf{X}^{(0)}: 1--2--3--4--5--1$$

sorrendben megy végig a városokon. Ekkor $z(\mathbf{X}^{(0)})=20$.

$$\mathbf{C}=\begin{pmatrix} & 3 & 4 & 1 & 2\\ & & 2 & 5 & 3\\ & & & 6 & 2\\ & & & & 7\\ & & & & \end{pmatrix}$$

A kiindulási körút szomszédjait tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
0	$(1,2),(3,4)$	$1-3-2-4-5-1$	20
0	$(1,2),(4,5)$	$1-4-3-2-5-1$	14
0	$(2,3),(4,5)$	$1-2-4-3-5-1$	18
0	$(2,3),(1,5)$	$1-2-5-4-3-1$	23
0	$(3,4),(1,5)$	$1-2-3-5-4-1$	15

Tehát

$$\mathbf{X}^{(1)} : 1 - -4 - -3 - -2 - -5 - -1, \;\; z(\mathbf{X}^{(1)}) = 14.$$

Az eljárás következő iterációs lépésében számolt szomszédokat tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
1	$(1,4),(2,3)$	$1-3-4-2-5-1$	20
1	$(1,4),(2,5)$	$1-2-3-4-5-1$	20
1	$(3,4),(2,5)$	$1-4-2-3-5-1$	12
1	$(1,5),(3,4)$	$1-3-2-5-4-1$	17
1	$(2,3),(1,5)$	$1-4-3-5-2-1$	15

Ekkor

$$\mathbf{X}^{(2)} : 1 - -4 - -2 - -3 - -5 - -1, \;\; z(\mathbf{X}^{(2)}) = 12.$$

Az eljárás által a következő iterációs lépésben számolt szomszédokat tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
2	$(1,4),(2,3)$	$1-2-4-3-5-1$	18
2	$(1,4),(3,5)$	$1-3-2-4-5-1$	20
2	$(2,4),(3,5)$	$1-4-3-2-5-1$	14
2	$(2,4),(1,5)$	$1-4-5-3-2-1$	15
2	$(2,3),(1,5)$	$1-4-2-5-3-1$	15

Ekkor $\mathbf{X}^{(2)}$ minden $\bar{\mathbf{X}}$ szomszédjára $z(\mathbf{X}^{(2)}) \leq z(\bar{\mathbf{X}})$ teljesül, így vége az eljárásnak, és a heurisztika által kapott megoldás az $(1,4,2,3,5,1)$ körút.

Kézenfekvő a 2-optimális eljárás olyan általánosítása (ld. [35], [127]), amikor a tekintett körútból k számú páronként nem szomszédos élet törlünk,

majd a keletkező utakból k él felvételével új körutat alakítunk ki, ahol $k > 1$ rögzített egész. Az ilyen heurisztikát *k-optimális eljárásnak* nevezzük.

A fejezet hátralevő részében olyan TSP feladatokat vizsgálunk, amelyekben a költségmátrix szimmetrikus és teljesül rá a háromszögegyenlőtlenség. Az ilyen mátrixok esetére ismertetünk egy 3/2-approximációs heurisztikát.

Az eljárás ismertetéséhez a probléma gráfelméleti reprezentációját használjuk. A problémához egy teljes (bármely két pont között megy él) irányítatlan hálózatot rendelünk, amelynek csúcsai a városoknak felelnek meg, és az éleken a hosszak a két várost összekötő út hosszai. A feladat ekkor az, hogy megtaláljuk azt az összes ponton átmenő kört, amelyre a körben szereplő élek hosszainak összege minimális.

Az eljárás megadásához szükségünk lesz a következő definícióra. Egy Euler féle multigráf Euler körének a *rövidítésén* pontoknak azt a sorozatát értjük, amelyet úgy kapunk, hogy a körben minden csúcsnak csak az első előfordulását tartjuk meg, amennyiben többször is szerepel a további előfordulásait töröljük a pontok listájából.

11.2. példa. A 2.5. példában tekintett multigráfra meghatározott Euler kör pontok sorozatával a következőképpen írható le: $(1, 6, 5, 4, 3, 2, 5, 2, 1)$. Az Euler kör rövidítése az $(1, 6, 5, 4, 3, 2)$ pontsorozatot adja meg.

Egy Euler kör rövidítése pontok egy sorbarendezését adja meg, amelyből egy körutat kapunk, ha a kezdőpontot összekötjük a végponttal. Amennyiben a pontok multigráfjához tartozó élekre teljesül a háromszögegyenlőtlenség, akkor az Euler körben szereplő élek súlyainak összegére és a rövidítéshez tartozó körútban szereplő élek hosszainak összegére teljesül a következő állítás.

11.1. segédtétel. *A rövidítéshez tartozó körútban az élek súlyainak összege legfeljebb annyi, mint az Euler körben az élek súlyainak összege.*

Bizonyítás. Legyen $(p_1, \ldots, p_n)$ a rövidítésben a pontok sorozata. A továbbiakban a p_{n+1} pontot a p_1 pontnak feleltetjük meg. Jelölje az Euler körben a p_i és p_{i+1} pontok első előfordulásai közötti pontokat $q_{i1}, \ldots, q_{ik_i}$, ahol a pontok ezen sorozata üres is lehet.

A háromszögegyenlőtlenséget alkalmazva többször egymás után adódik, hogy minden i-re teljesül $c_{p_i p_{i+1}} \leq c_{p_i q_{i1}} + \sum_{j=1}^{k_i - 1} c_{q_{ij}, q_{i,j+1}} + c_{q_{ik_i} p_{i+1}}$. Ezen értékeket összegezve adódik a segédtétel állítása.

A Euler körök rövidítésének alkalmazásával megadhatjuk N. Christofides heurisztikus algoritmusát.

Christofides eljárása ([34])

- *1. lépés.* Határozzunk meg a feladatot leíró $\mathcal{G}$ hálózatban egy minimális hosszúságú feszítőfát Kruskal algoritmusával. Jelölje ezt a fát $\mathcal{T}$.
- *2. lépés.* Legyen $\mathcal{G}'$ az a részgráfja $\mathcal{G}$-nek, amelynek a pontjai $\mathcal{T}$ páratlan fokszámú pontjai, és az élei $\mathcal{G}$ azon élei, amelyek ezen pontok között mennek. Határozzuk meg a 3.3. fejezetben leírt algoritmus alapján $\mathcal{G}'$ egy minimális költségű teljes párosítását. Egészítsük ki a párosításban szereplő élekkel a $\mathcal{T}$ fát.
- *3. lépés.* Az így kapott multigráfban határozzunk meg egy Euler kört.
- *4. lépés.* Vegyük a kapott Euler kör rövidítését, a rövidítéshez tartozó körút a heurisztikával előállított megoldás.

Elsőként igazoljuk, hogy az eljárás végrehajtható.

11.2. segédtétel. *A Christofides eljárásban megadott lépések mindegyike végrehajtható, és az eredmény egy körút lesz.*

Bizonyítás. Mivel a kiindulási gráf egy teljes gráf, ezért összefüggő, így Kruskal algoritmusa valóban egy feszítőfát ad eredményül. Mivel egy gráfban a csúcsok fokszámainak összege az élek számának kétszerese (minden élt beszámolunk mindkét végpontjánál), ezért a fokszámok összege páros. Másrészt ha a fokszámok összege páros, akkor páros azon pontoknak a száma, amelyeknek páratlan a fokszáma. Így a 2. lépésben tekintett $\mathcal{G}'$ gráfnak páros számú csúcsa van, továbbá a gráf teljes, így a párosítási feladatnak van optimális megoldása. A megoldásban szereplő éleket hozzávéve $\mathcal{T}$-hez egy olyan összefüggő multigráfhoz jutunk, amelyben minden csúcs fokszáma páros (azon pontok fokszáma, amelyeké páratlan volt 1-gyel növekszik). Tehát a multigráf kielégíti a 2.1. tétel feltételeit, így van Euler köre, következésképpen a 3. lépés végrehajtható. Mivel az Euler kör rövidítése során valóban egy körutat kapunk, azért a 4. lépés után egy lehetséges megoldáshoz jutunk. Fontos megjegyeznünk, hogy mivel minden lépés polinomiális időben végrehajtható, ezért a teljes algoritmus is.

Az algoritmus egy optimálishoz közeli megoldást ad, miként azt a következő állítás mutatja.

11.1. tétel. *Christofides heurisztikus algoritmusa* 3/2*-approximációs algoritmus.*

Bizonyítás. Tekintsünk egy tetszőleges $TSP(\mathbf{C})$ problémát. Jelölje az eljárás első lépése során kapott $\mathcal{T}$ feszítőfa hosszát $c(\mathcal{T})$. Jelölje a második lépésben megkonstruált teljes párosításra a párosításban szereplő élek összhosszát $c(M)$. Végül jelöljük a probléma optimális megoldásának költségét $\mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$-vel.

Elsőként vegyük észre, hogy egy körútból elhagyva bármely élét egy feszítőfát kapunk, így a minimális hosszúságú feszítőfa hossza nem nagyobb, mint az optimális körút költsége, azaz $c(\mathcal{T}) \leq \mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$.

Most vizsgáljuk a párosítás költségét. Legyenek $p_1, p_2, \ldots, p_{2j}$ a $\mathcal{T}$ feszítőfa páratlan fokszámú pontjai. Az általánosság megszorítása nélkül feltehetjük, hogy ezek a pontok ilyen sorrendben helyezkednek el a tekintett $TSP(\mathbf{C})$ probléma optimális megoldást adó körútjában is, hiszen ezt a pontok átindexelésével elérhetjük. Másrészt ekkor a 11.1. segédtétel bizonyításához hasonlóan a háromszögegyenlőtlenség többszöri alkalmazásával azt kapjuk, hogy a $c_{p_i,p_{i+1}}$ távolság legfeljebb annyi, mint az optimális körútban a p_i és a p_{i+1} pontokat összekötő útszakaszban az élek hosszainak összege. Következésképpen

$$\sum_{i=1}^{2j-1} c_{p_i,p_{i+1}} + c_{p_{2j},p_1} \leq \mathrm{opt}_{\mathrm{TSP}}(\mathbf{C}).$$

Most vegyük a következő teljes párosításait a $p_1, p_2, \ldots, p_{2j}$ pontoknak. Legyen M_1 a (p_{2j}, p_1) és a (p_{2i}, p_{2i+1}), $(i = 1, \ldots, j-1)$ pontpárokat összekötő élek halmaza, M_2 pedig a (p_{2i+1}, p_{2i+2}), $(i = 0, \ldots, j-1)$ pontpárokat összekötő élek halmaza. Ekkor a két párosítás költségének az összege:

$$\sum_{i=1}^{2j-1} c_{p_i,p_{i+1}} + c_{p_{2j},p_1}.$$

Vegyük észre, hogy a kapott összeg pontosan a fenti egyenlőtlenség baloldala, ezért nem nagyobb, mint $\mathrm{opt}_{TSP}(\mathbf{C})$. Ezen észrevételek alapján adódik, hogy a két párosítás közül arra, amelynek kisebb a költsége, ez a költség legfeljebb $\mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})/2$. Másrészt ebből az M párosításra következik, hogy $c(M) \leq \mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})/2$, mivel M minimális költségű teljes párosítása a tekintett pontoknak.

A $c(T)$ és $c(M)$ értékekre adott becsléseink alapján adódik, hogy az eljárás 3. lépésében megkonstruált Euler körben az élek hosszainak az összege legfeljebb $\frac{3}{2}\mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$, amiből a 11.1. segédtétel alapján adódik, hogy a heurisztika által szolgáltatott megoldásban is legfeljebb $\frac{3}{2}\mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$ az élek hosszainak az összege, amivel a tétel állítását igazoltuk.

A fejezet lezárásaként a következő példán mutatjuk be Christofides algoritmusát.

11.3. példa. Tekintsük azt az 5 városos utazó ügynök feladatot, amelynek költségmátrixa az alábbi $\mathbf{C}$ mátrix

$$\mathbf{C} = \begin{pmatrix} & 2 & 3 & 3 & 5 \\ & & 4 & 4 & 2 \\ & & & 3 & 3 \\ & & & & 6 \\ & & & & \end{pmatrix}$$

Elsőként hajtsuk végre Kruskal algoritmusát. Az algoritmus a következő éleket választja ki a megadott sorrendben $(1,2),(2,5),(1,3),(1,4)$. A kapott feszítőfa hossza 10. A feszítőfában a páratlan fokszámú pontok $1,3,4,5$. Megoldva az ezen pontokból és a közöttük futó élekből álló gráfra a párosítási problémát, azt kapjuk, hogy a minimális költségű teljes párosítás az $(1,4)$ és $(3,5)$ éleket tartalmazza, és a párosítás költsége 6. Tehát a 3. lépésben vizsgált multigráfnak az élei $(1,2),(1,3),(1,4),(1,4),(2,5),(3,5)$. Az Euler kört kereső algoritmus a következő pontsorozatot adja vissza $1,2,5,3,1,4,1$. Ezen Euler körnek a rövidítése az $1,2,5,3,4$ sorrendje a pontoknak, így a Christofides algoritmus által adott körút az $1-2-5-3-4-1$, amely körútnak a költsége 13.

12. Az utazó ügynök problémával rokon feladatok

Az előzőek folyamán többféleképp is formalizáltuk a hozzárendelési feladatot. Most a korábbiakkal ekvivalens olyan formalizmust vezetünk be, amely lehetővé teszi a jelen fejezetben vizsgált modellek egységes formalizálását.

Tekintsünk egy $A(\mathbf{C})$ hozzárendelési feladatot az $a_1, \ldots, a_n$ dolgozókkal és a $b_1, \ldots, b_n$ munkákkal. Tudjuk, hogy a lehetséges megoldások pontosan azon $\{0,1\}$ feletti $\bar{\mathbf{X}}^{n\times n}$ mátrixok, amelyeknek minden sora és minden oszlopa pontosan 1 db 1-est tartalmaz. Ekkor $\bar{x}_{ij} = 1$ azt jelenti, hogy a b_j munkát rendeljük az a_i dolgozóhoz. Így $\bar{\mathbf{X}}$ meghatározza az $\{a_1, \ldots, a_n\}$ halmaz egy kölcsönösen egyértelmű leképezését a $\{b_1, \ldots, b_n\}$ halmazra. Most vegyük észre, hogy amennyiben $\bar{\mathbf{X}}$-ra az $a_1, \ldots a_n$ elemek indexeinek feleltetjük meg a megfelelő $b_1, \ldots, b_n$ elemek indexeit, akkor az $\{1, \ldots, n\}$ halmaz egy önmagára való kölcsönösen egyértelmű φ leképezését kapjuk, amit az $\{1, \ldots, n\}$ *halmaz permutációjának* nevezünk. (Például, ha $\bar{x}_{11} = \bar{x}_{22} = 0, \bar{x}_{12} = \bar{x}_{21} = 1$, akkor $\varphi(1) = 2$ és $\varphi(2) = 1$, azaz φ az $(1,2)$ ciklikus permutáció.) Egyszerűen belátható, hogy a vázolt módon $A(\mathbf{C}^{n\times n})$ lehetséges megoldásainak L halmaza kölcsönösen egyértelmű módon leképezhető az $\{1, \ldots, n\}$ halmaz permutációnak halmazára. Jelöljük az $\{1, \ldots, n\}$ halmaz permutációnak halmazát $\mathcal{P}$-vel. Felhasználva $\bar{\mathbf{X}}$ és a hozzárendelt $\varphi_{\bar{\mathbf{X}}}$ permutáció definícióit, egyszerűen belátható, hogy $A(\mathbf{C})$ minden $\bar{\mathbf{X}}$ lehetséges megoldására

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\bar{x}_{ij} = \sum_{i=1}^{n} c_{i,\varphi_{\bar{\mathbf{X}}}(i)}$$

teljesül. A fentiekből egyszerűen következik, hogy az $A(\mathbf{C})$ hozzárendelési feladat egy ekvivalens leírása a következő feladat:

$$\begin{array}{c} \varphi \in \mathcal{P} \\ \hline \sum_{i=1}^{n} c_{i,\varphi(i)} \to \min \end{array} \qquad (12.1)$$

A 10. és 11. fejezetekben már észrevettük és felhasználtuk, hogy a hozzárendelési feladat optimális megoldása vagy egy körutat vagy diszjunkt körutakból álló gráfot szolgáltat. Ennek az észrevételnek a megfelelője az az algebrából ismert tény, hogy minden permutáció vagy ciklikus permutáció vagy diszjunkt ciklusok egyesítéséből áll. Ha $\mathcal{P}$ azon permutációit vesszük, amelyek egyetlen ciklusból állnak, azaz az $\{1, \ldots, n\}$ halmaz ciklikus permutációit, akkor pontosan az összes lehetséges körutat kapjuk meg. Így a $TSP(\mathbf{C})$ feladat felírható az alábbi formában.

$$\begin{array}{c} \varphi \in \mathcal{P} \\ \varphi \text{ egyetlen ciklusból áll} \\ \hline \sum_{i=1}^{n} c_{i,\varphi(i)} \to \min \end{array} \tag{12.2}$$

Az eddig használt terminológiában az utazó ügynök problémával rokon, két további problémát is egyszerűen le lehet írni. Az első annyiban különbözik a TSP-től, hogy adott egy $1 \leq p \leq n$ egész, és a városokat legfeljebb p számú ügynök alkalmazásával kell végiglátogatni úgy, hogy minden várost a használt ügynökök egyike pontosan egyszer érintsen, minden várost csak egy ügynök látogasson meg, továbbá a használt ügynökök által megtett összes út hossza minimális legyen. A probléma az alábbi optimumszámítási modellel írható le:

$$\begin{array}{c} \varphi \in \mathcal{P} \\ \varphi \text{ legfeljebb } p \text{ ciklusból áll} \\ \hline \sum_{t=1}^{n} c_{t,\varphi(t)} \to \min \end{array} \tag{12.3}$$

Ez a feladat az irodalomban (ld. pl. [26], [123]) TSP *with p traveling salesmen* néven ismeretes és pTSP-nek rövidítik. Mi is ezt a rövidítést fogjuk használni.

A másik rokon feladat az irodalomban *Hamiltonian p-Median Problem* (HpMP röviden) néven ismeretes (ld. [24], [31], [74]). Ez annyiban különbözik a pTSP-től, hogy a p ügynök mindegyikét kötelező használni. Ennek megfelelően a következő optimumszámítási modellel írható le:

$$\begin{array}{c} \varphi \in \mathcal{P} \\ \varphi \text{ pontosan } p \text{ ciklusból áll} \\ \hline \sum_{t=1}^{n} c_{t,\varphi(t)} \to \min \end{array} \tag{12.4}$$

A pTSP és HpMP optimumszámítási modellek olyan gyakorlati kiszolgálási feladatokat írnak le, mint a különböző termékek begyűjtésének megszervezése, szállító eszközök vezénylése, stb.

A továbbiakban a $\mathbf{C}$ költségmátrixú $A(\mathbf{C})$, $TSP(\mathbf{C})$, $p\mathrm{TSP}(\mathbf{C})$, $\mathrm{H}p\mathrm{MP}(\mathbf{C})$ feladatok optimum értékeire rendre az $\mathrm{opt}_{\mathrm{A}}(\mathbf{C})$, $\mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$, $\mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C})$, és $\mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{C})$ jelöléseket használjuk.

Elsőként a három utazó ügynök feladat optimumértékeit illetően adunk meg egy érdekes összehasonlítást. Ehhez használni fogjuk a következő definíciót. Legyen φ az $\{1,\ldots,n\}$ halmaz egy permutációja, és $(i_1,\ldots,i_k)$ a φ permutáció egy ciklusa. Az $(i_1,\ldots,i_k)$ ciklust *erősen piramidálisnak* nevezzük, ha $i_1 < \ldots < i_k$. (Például $n = 6$-ra az $(1,4,2)$ és $(3,5,6)$ ciklusokból álló permutáció második ciklusa erősen piramidális.)

Tekintsük ezek után az optimumok értékeit. Mivel a pTSP mind a TSP-nek, mind a HpMP-nek relaxációja, rögtön adódik, hogy bármely $\mathbf{C}$ költségmátrix mellett $\mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C}) \leq \mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$ és $\mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C}) \leq \mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{C})$ érvényes. Az általánosan érvényes relációnál szorosabb összefüggés teljesül, ha a $\mathbf{C}$ költségmátrixra érvényes a háromszögegyenlőtlenség, azaz minden $i,j,k \in \{1,\ldots,n\}$-re $c_{ij} + c_{jk} \geq c_{ik}$. Erre az esetre [96]-ban igazolást nyert, hogy

12.1. segédtétel. $\mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{C}) \leq p \cdot \mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C}) \leq p \cdot \mathrm{opt}_{\mathrm{TSP}}(\mathbf{C})$.

Ami kissé meglepő, hogy [96]-ban azt is sikerült igazolni, hogy ezek a korlátok élesek, nevezetesen érvényes a következő állítás.

12.1. tétel ([96]). *Minden pozitív n egészre van olyan $\mathbf{D}_n$ költségmátrix, amelyre teljesül a háromszögegyenlőtlenség és minden $1 \leq p \leq n$-re*

$$\mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{D}_n) = p \cdot \mathrm{opt}_{p\mathrm{TSP}}(\mathbf{D}_n) = p \cdot \mathrm{opt}_{\mathrm{TSP}}(\mathbf{D}_n).$$

Bizonyítás. Legyen n egy pozitív egész és definiáljuk a $\mathbf{D}_n$ $n \times n$-es mátrixot a következők szerint. Legyen

$$d_{ij} = \begin{cases} n - i + j, & \text{ha } i \geq j, \\ j - i & \text{különben.} \end{cases}$$

Akkor az alábbi tények egyszerűen beláthatók.

(1) $\mathbf{D}_n$-re teljesül a háromszögegyenlőtlenség,

(2) bármely ciklus hossza nem kisebb, mint n,

(3) bármely erősen piramidális ciklus hossza n-nel egyenlő.

Most legyen $1 \leq p \leq n$ egy tetszőleges egész. Ha $p = 1$, akkor az állítás nyilvánvalóan teljesül. Ezek után tegyük fel, hogy $p \neq 1$. Akkor a (2) és (3)

állításokból következik, hogy bármely p számú erősen piramidális ciklusból álló φ permutáció hossza pn és ezek a permutációk optimális megoldásai a HpMP($\mathbf{D}_n$) feladatnak. Másrészt az $(1, 2, \ldots, n)$ ciklus hossza n, így $d_{ij} \geq 1$ miatt ez optimális megoldása a $TSP(\mathbf{D}_n)$ feladatnak. Következésképpen

$$\mathrm{opt}_{\mathrm{HpMP}}(\mathbf{D}_n) = p \cdot n = p \cdot \mathrm{opt}_{\mathrm{TSP}}(\mathbf{D}_n),$$

amivel a 12.1. tételt igazoltuk.

Ismeretes, hogy a TSP (ld. [67]) probléma NP-nehéz. Mivel a $p = 1$ esetben mind a pTSP mind pedig a HpMP probléma a TSP problémára redukálódik, ezért ezek a problémák is NP-nehezek. Szintén igazolást nyert, hogy tetszőleges p konstans mellett a HpMP (ld. [31]) NP-nehéz. Másrészt egyszerűen adódik, hogy tetszőleges p konstans mellett a pTSP probléma is NP-nehéz, ugyanis teszőleges TSP probléma megoldása visszavezethető egy alkalmas pTSP probléma megoldására felvéve $p - 1$ további pontot, melyek távolsága egymástól és az eredeti pontoktól egy elegendően nagy érték. Következésképp mindhárom esetben létjogosultsága van szuboptimális lehetséges megoldásokat szolgáltató heurisztikáknak. Az előző fejezetben a TSP problémára megismerkedtünk több, különböző heurisztikus eljárással. Ezek közül az egyik a 2-patching eljárás volt. A fejezet további részében megmutatjuk, hogy az általános pTSP feladatra a 2-patching eljárás egyszerű adaptálásával kaphatunk egy heurisztikát, míg az általános HpMP feladatra a 2-patching eljárás egy elég kézenfekvő fejlesztésével nyerhető heurisztika. Az ismertetésre kerülő mindkét heurisztikára bemutatunk egy-egy empirikus analízist is, amelyek mutatják a tekintett mintákra a heurisztikákkal előállított lehetséges megoldások jóságát.

2-patching heurisztika pTSP($\mathbf{C}$)-re

(1) Oldjuk meg az $A(\mathbf{C})$ hozzárendelési feladatot. Legyen φ $A(\mathbf{C})$ optimális megoldása és $\vartheta = \varphi$. Továbbá legyen ϑ diszjunkt ciklusainak a száma k.

(2) Ha $k \leq p$, akkor vége az eljárásnak, ϑ a pTSP($\mathbf{C}$) egy közelítő megoldása. Ellenkező esetben folytassuk a (3) lépéssel az eljárást.

(3) Rendezzük ϑ ciklusait a tartalmazott pontok száma szerint csökkenő sorrendbe. Hajtsunk végre az első két cikluson egy 2-patching műveletet. Az új permutációt jelölje ϑ, csökkentsük k értékét 1-gyel, majd folytassuk az eljárást a (2) lépéssel.

A korábbi empirikus analíziseket követve, az ismertetett heurisztikára a [95] dolgozatban végrehajtottunk egy empirikus analízist. Ennek során n=100,150,200 feladatméretekre és $p = 2, 3, 5, 7, 10$ értékekre 100-100 költségmátrixot generáltunk véletlenszerűen egyenletes eloszlás mellett az [1,100] intervallumba eső egészekből. Minden generált $\mathbf{C}$ költségmátrixra meghatároztuk az $\mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C})$ optimumot, továbbá végrehajtottuk $p\mathrm{TSP}(\mathbf{C})$-re a 2-patching eljárás. Jelölje $K^*(\mathbf{C})$ a 2-patching eljárással nyert közelítő megoldáson felvett célfüggvényértéket. Az empirikus analízis során kapott $K^*(\mathbf{C})/\mathrm{opt}_{p\mathrm{TSP}}(\mathbf{C})$ hányadosok K^*/opt-tal jelölt átlagát valamint a heurisztikával nyert optimális megoldások s_n számát adja meg a 12.1. táblázat.

	$n = 100$		$n = 150$		$n = 200$	
p	$\mathrm{K}^*/\mathrm{opt}$	s_n	$\mathrm{K}^*/\mathrm{opt}$	s_n	$\mathrm{K}^*/\mathrm{opt}$	s_n
2	1.0515	12	1.0359	13	1.0258	7
3	1.0284	34	1.0263	21	1.0257	16
5	1.0133	68	1.0108	56	1.0105	56
7	1.0023	96	1.0017	92	1.0018	83
10	1.0000	100	1.0000	100	1.0000	100

12.1. táblázat.

A HpMP feladat egy szuboptimális lehetséges megoldásának meghatározására több heurisztikus eljárás került kidolgozásra a [74] dolgozatban. Itt most egy olyan heurisztikus eljárást idézünk fel [96]-ból, amely részben a 2-patching technikán alapul. Az eljárás alapötlete az, hogy a hozzárendelési feladat optimális megoldásából indulunk ki. Ha az optimális megoldásban a ciklusok k száma megegyezik p-vel, akkor az illető permutáció optimális megoldása HpMP-nek is. Ha $k > p$, akkor ciklusok 2-patching műveleteinek egy sorozatával készítünk egy lehetséges megoldást. Ha pedig $k < p$, akkor ciklusok két ciklussá történő vágásával növeljük a ciklusok számát addig, amíg a ciklusok száma el nem éri p-t. Az eljárás *Cutting and Patching Method* néven került bevezetésre, amire a továbbiakban, mint *vágási és 2-patching eljárásra* fogunk hivatkozni. Az eljáráshoz a ciklusok vágását kell pontosítani. Ehhez legyen adott egy nem loop $(i_0, i_1, \ldots, i_{r-1})$ ciklus. Továbbá legyenek $0 \leq s < r$ és $0 \leq t \leq \lfloor (r-2)/2 \rfloor$ tetszőleges egészek. Tekintsük a ciklus által meghatározott irányított részkörutat. Elindulva ezen a részkörúton az i_s csúcsból és bejárva a részkörút csúcsait i_v-ig, ahol $v = s + t (\mathrm{mod}\ r)$, majd ehhez az irányított úthoz hozzávéve az (i_v, i_s) élet egy irányított részkörutat kapunk. A másik részkörutat úgy képezzük, hogy a tekintett ciklus által meghatározott körút i_u csúcsából

indulunk, ahol $u = v + 1 (\mathrm{mod}\ r)$, és bejárjuk a csúcsokat i_w-ig, ahol $w = s + r - 1 (\mathrm{mod}\ r)$, majd felvesszük az (i_w, i_u) élet. A kapott két részkörút két diszjunkt ciklust eredményez, amelyek két kivétellel ugyanazon éleket tartalmazzák, mint a tekintett ciklus. A ciklus szétvágásának költségeként tekinthető az új részkörutak hosszai összegének a tekintett ciklushoz tartozó részkörút hosszától való eltérése, ami

$$c_{i_v,i_s} + c_{i_w,i_u} - c_{i_v,i_u} - c_{i_w,i_s}.$$

Most minden lehetséges s és t egészre kiszámítva a fenti költséget, majd képezve a kapott költségek minimumát, megkapjuk, hogy ilyen típusú vágások közül, mely vágás jár minimális költséggel. Ezt a költséget a tekintett *ciklus vágási költségének* nevezzük. Ezek után a HpMP feladatra az alábbi heurisztikus eljárást lehet alkalmazni.

Vágási és 2-patching eljárás ([96])

(1) Oldjuk meg az A($\mathbf{C}$) hozzárendelési feladatot. Legyen φ A($\mathbf{C}$) optimális megoldása és $\vartheta = \varphi$. Továbbá legyen ϑ diszjunkt ciklusainak a száma k.

(2) Ha $k = p$, akkor vége az eljárásnak, ϑ a HpMP($\mathbf{C}$) egy közelítő megoldása. Ha $k < p$, akkor a (4) lépés következik. Ellenkező esetben folytassuk az eljárást a (3) lépéssel.

(3) Válasszuk ki ϑ két olyan ciklusát, amelyekre minimális a 2-patching költség, majd hajtsunk végre ezen a két cikluson egy 2-patching műveletet. A kapott új ciklushoz vegyük hozzá ϑ maradék ciklusait, és az így előálló permutáció legyen ϑ. Csökkentsük k értékét 1-gyel és folytassuk az eljárást a (2) lépéssel.

(4) Válasszuk ki ϑ egy olyan ciklusát, amelynek minimális a vágási költsége. Vágjuk szét ezt a ciklust a minimális vágási költségnek megfelelően. A tekintett ciklust helyettesítsük ϑ-ban a vágással kapott két új ciklussal és az így előálló új permutáció legyen ϑ. Növeljük k értékét 1-gyel és folytassuk az eljárást a (2) lépéssel.

A Karp-féle eljárás empirikus analíziséhez hasonló empirikus analízist tartalmaz a vágási és 2-patching eljárás vizsgálatára a [96] dolgozat. Az analízisben n=50,100,150 feladatméretekre és $p = 1, 2, 3, 4, 5, 7, 10$ értékekre 100-100 költségmátrix került generálásra véletlenszerűen egyenletes eloszlás

mellett az [1,100] intervallumba eső egészekből. Minden generált $\mathbf{C}$ költségmátrixra meghatározásra került $\mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{C})$, továbbá végrehajtásra került HpMP($\mathbf{C}$)-re a vágási és 2-patching eljárás. Jelölje V($\mathbf{C}$) a vágási és 2-patching eljárással nyert közelítő megoldáson felvett célfüggvényértéket. Az empirikus analízis során kapott V($\mathbf{C}$)/$\mathrm{opt}_{\mathrm{H}p\mathrm{MP}}(\mathbf{C})$ hányadosok V/opt-tal jelölt átlagát adja meg a 12.2. táblázat.

	$n = 50$	$n = 100$	$n = 150$
p	V/opt	V/opt	V/opt
1	1.099	1.098	1.076
2	1.018	1.030	1.028
3	1.002	1.015	1.019
4	1.006	1.003	1.010
5	1.002	1.003	1.008
7	1.010	1.032	1.013
10	1.107	1.064	1.033

12.2. táblázat

13. Halmazlefedési feladat

A címben szereplő feladat hasonlóan a TSP-hez a kombinatorikus optimalizálás egy sok érdeklődést kiváltó, igen nevezetes problémája. Elsőként a probléma egy általános megfogalmazását adjuk meg, majd bemutatunk néhány gyakorlati alkalmazást.

Halmazlefedési probléma

Adott egy $I = \{1, \ldots, n\}$ halmaz és I nemüres részhalmazainak egy $P_1, \ldots, P_m$ rendszere a $c_1, \ldots, c_m$ pozitív súlyokkal. Határozzuk meg I-nek egy minimális súlyú, a $P_1, \ldots, P_m$ részhalmazokból felépülő fedőrendszerét. (Itt fedőrendszer súlyán a benne szereplő részhalmazok súlyainak összegét értjük.)

Vezessük be a következő jelöléseket. Tetszőleges $i \in I$, $1 \leq j \leq m$ indexekre legyen

$$a_{ij} = \begin{cases} 1, & \text{ha } i \in P_j, \\ 0 & \text{különben,} \end{cases}$$

továbbá

$$x_j = \begin{cases} 1, & \text{ha } P_j \text{ eleme a fedőrendszernek,} \\ 0 & \text{különben.} \end{cases}$$

Akkor a feladat az alábbi *halmazlefedési probléma* néven ismert optimumszámítási modellel írható le.

$$\sum_{j=1}^{m} a_{ij} x_j \geq 1 \quad (i = 1, \ldots, n)$$

$$x_j \in \{0, 1\} \quad (j = 1, \ldots, m), \quad (\mathbf{c} > \mathbf{0}) \tag{13.1}$$

$$\sum_{j=1}^{m} c_j x_j = z \to \min$$

Vegyük észre, hogy (13.1)-nek akkor és csak akkor létezik lehetséges megoldása, ha minden egyenlőtlenség baloldala tartalmaz legalább egy 0-tól különböző együtthatót. A továbbiakban ezt minden hivatkozás nélkül

feltételezzük. Másrészt a lehetséges megoldások L halmazára $L \subseteq \{0,1\}^m$ teljesül, így L véges, de akkor létezik a (13.1) feladatnak optimális megoldása.

A modellel kapcsolatosan fontos megemlíteni, hogy a $\mathbf{c} > \mathbf{0}$ feltétel nem jelent tényleges megszorítást. Valóban, $c_j \leq 0$ $(j \in J)$ esetén képezzük az

$$\bar{x}_j = \begin{cases} 1, & \text{ha } j \in J, \\ 0 & \text{különben,} \end{cases}$$

komponensekkel rendelkező $\bar{\mathbf{x}}$ vektort. Ha $\bar{\mathbf{x}}$ lehetséges megoldás, akkor egyben optimális megoldás is. Ellenkező esetben egyszerűen belátható, hogy (13.1) bármely $\tilde{\mathbf{x}}$ optimális megoldásához létezik egy $\bar{\mathbf{x}}$ lehetséges megoldás a következő tulajdonságokkal:

(1) $z(\bar{\mathbf{x}}) = z(\tilde{\mathbf{x}})$,

(2) $\bar{x}_j = 1$, ha $j \in J$.

Ebből viszont következik, hogy (13.1)-ben minden $j \in J$ indexre az x_j változónak 1 értéket adva, majd az előálló (pozitív súlyokkal rendelkező) problémát megoldva, megkapjuk az eredeti feladat egy optimális megoldását.

Egy másik modellt kapunk, ha a kitűzött problémában nem fedőrendszert, hanem az I halmaz minimális súlyú, a $P_1, \ldots, P_m$ halmazokból felépülő osztályozását keressük. Ebben az esetben *halmazosztályozási problémáról* beszélünk. A feladatot leíró optimumszámítási modell csupán annyiban tér el (13.1)-től, hogy mindenhol egyenlőtlenség helyett egyenlőséget követelünk meg, azaz

$$\sum_{j=1}^{m} a_{ij} x_j = 1 \quad (i = 1, \ldots, n)$$

$$x_j \in \{0,1\} \quad (j = 1, \ldots, m), \quad (\mathbf{c} > \mathbf{0}) \tag{13.2}$$

$$\sum_{j=1}^{m} c_j x_j = z \to \min$$

A $\mathbf{c} > \mathbf{0}$ kikötése itt sem jelent lényeges megszorítást, ugyanis mivel a $P_1, \ldots, P_m$ részhalmazok egyike sem üres, ezért az $\mathbf{A}$ mátrix minden oszlopvektora tartalmaz legalább egy darab 1-est. Másrészt $c_j \leq 0$ esetén véve

egy olyan egyenletet, amelyben x_j együtthatója 1, és ezen egyenlet $(-c_j+1)$-szeresét hozzáadva a célfüggvény egyenletéhez, az egyenlőség miatt az új és a régi célfüggvény megegyezik a lehetséges megoldások halmazán. A fentieket végrehajtva minden $c_j \leq 0$ célfüggvényegyütthatóra, az előálló feladat már rendelkezik a $\mathbf{c} > \mathbf{0}$ tulajdonsággal.

A lehetséges megoldások létezését illetően nem olyan egyszerű a kérdés megválaszolása, mint a (13.1) feladat esetében. Mielőtt erre rátérnénk, megadjuk a modell néhány gyakorlati alkalmazását.

Információ kigyűjtés

Adott m féle file ugyanazon anyagról különböző információtartalmakkal. Egy olyan új file-t akarunk képezni, amely n féle információféleséget tartalmaz és ezek mindegyike megtalálható az előzőek valamelyikében. Mely filekat másoljuk össze, hogy a kívánt új file hossza minimális legyen.

A fenti alkalmazást R. Day ismertette 1965-ben a [41] dolgozatban. Ha kikötjük, hogy az új file-ban az előírt információféleségek mindegyike egynél többször nem szerepelhet, akkor a probléma egy halmazosztályozási feladatot eredményez.

Termelésszervezés

Adott n elvégzendő feladat és m olyan vállalat, amelyek mindegyike a fenti feladatok közül bizonyosakat el tud végezni. Válasszunk ki az adott vállalatok közül minimális számút úgy, hogy ezek együttesen valamennyi feladatot képesek legyenek ellátni.

Ha kikötjük, hogy a feladatok elvégzésére kiválasztott vállalatok között nem lehet olyan kettő, amelyek ugyanazon feladatot tudják elvégezni, akkor halmazosztályozási problémához jutunk.

Játékszervezés

Egy társaságban mindenki hajlandó pingpongozni valakivel, de nem szükségképpen mindenkivel. (A hajlandóság kölcsönös.) Hogyan kell minimális számú játszmát lejátszani úgy, hogy mindenki legalább egyszer játsszon.

Ebben az esetben a fedőrendszer elemei olyan kételemű halmazok, amelyek az egymással játszani hajlandó személyeket tartalmazzák. Ha kikötjük,

hogy mindenki legfeljebb egyszer játszhat, akkor halmazosztályozási problémához jutunk, melynek megoldása a társaság egy párosítását eredményezi.

Kiszolgálási feladat

Adott egy raktár és n számú kliens, melyek mindegyikének van valamilyen megrendelése és ez a raktárból kiszolgálható. A raktárból szállításokkal elégítik ki a megrendeléseket. Egy szállítással legfeljebb k számú kliens elégíthető ki, és egy kliens megrendelése nem bontható meg, azt egy szállítmányon belül kell lerendezni. Határozzuk meg a megrendelések egy olyan kiszolgálását, amelynek teljes szállítási költsége minimális.

A fenti alkalmazást M. L. Balinski és R. R. Quandt ismertette 1964-ben a [12] dolgozatban. Vegyük észre, hogy egy klienst csak egyszer kell kiszolgálni, így a probléma (a halmazrendszer elemeinek alkalmas megválasztásával) egy halmazosztályozási feladatot eredményez.

Repülőgépjáratok személyzettel történő ellátása

Adott egy légitársaság, amelynek adott időszakban n számú járatot kell indítania (általában több különböző helyről) és rendelkezésére áll k számú teljes személyzet. (Bizonyos személyzetek csak bizonyos járatokra vezényelhetők.) Lássuk el az n járatot személyzettel úgy, hogy az ezzel járó összes költség (utazás, szállás, illetmény stb.) minimális legyen.

A feladatot M. Spitzer [167] vetette fel 1961-ben egy konferencián. A problémát leíró modell már nem annyira kézenfekvő, mint a megelőző esetekben, ezért valamelyest részletezzük.

Járatkombináción járatok egy olyan rendezett sorozatát értjük, melyet egy megfelelő személyzet folyamatosan ki tud szolgálni. Jelölje $R_t^{(j)}$ $(t = 1, \ldots, v_j)$ a j-edik személyzet által kiszolgálható járatkombinációk járatainak halmazait, továbbá legyen $I = \{1, \ldots, n\}$ a járatok halmaza. Akkor I egy olyan osztályozása, amelynek osztályai az $R_t^{(j)}$ $(t = 1, \ldots, v_j; j = 1, \ldots, k)$ részhalmazok közül kerülnek ki, biztosítja a járatok személyzettel történő ellátását. Előfordulhat viszont, hogy egy személyzetnek szimultán több járatkombinációt is ki kell szolgálnia. Ezt egy további feltétellel lehet kiküszöbölni. Nevezetesen, az $x_t^{(j)}$ változókra kikötjük, hogy $\sum_{t=1}^{v_j} x_t^{(j)} \leq 1$ $(j = 1, \ldots, k)$. Így minden személyzet legfeljebb egy járatkombinációt fog kiszolgálni.

Térjünk vissza ezek után a (13.2) feladat megoldhatóságára. A lehetséges

megoldás létezése összekapcsolható a (13.2) feladatnak egy alkalmas (13.1) típusú feladatra történő visszavezetésével. Ezt adja meg a következő, C. E. Lemke és társai [125] 1971-ben publikált állítása.

Konstruáljuk meg a (13.2) feladathoz az alábbi (13.3) feladatot:

$$\begin{gathered} \sum_{j=1}^{m} a_{ij}x_j \geq 1 \quad (i=1,\ldots,n) \\ x_j \in \{0,1\} \quad (j=1,\ldots,m) \qquad (13.3) \\ \hline \sum_{j=1}^{m} c'_j x_j = \tilde{z} \to \min \end{gathered}$$

ahol $c'_j = c_j + Rt_j$ $(j=1,\ldots,m)$, $R > \sum_{j=1}^{m} c_j$, $t_j = \sum_{i=1}^{n} a_{ij}$ $(j=1,\ldots,m)$. Ekkor érvényes a következő állítás.

13.1. tétel. *Ha a* (13.2) *feladatnak létezik lehetséges megoldása, akkor a* (13.2) *és* (13.3) *feladatok optimális megoldásai megegyeznek.*

Bizonyítás. Jelölje L és L' a (13.2) és (13.3) lehetséges megoldásainak a halmazát. Akkor $\emptyset \neq L \subseteq L'$, és mivel L' véges, ezért (13.3)-nak létezik optimális megoldása. Megmutatjuk, hogy tetszőleges $\bar{\mathbf{x}} \in L$ és $\mathbf{x}' \in L' \setminus L$ lehetséges megoldásokra $\tilde{z}(\bar{\mathbf{x}}) < \tilde{z}(\mathbf{x}')$ teljesül. Valóban, $\mathbf{x}' \in L' \setminus L$ miatt van olyan $i \in \{1,\ldots,n\}$ index, hogy $\sum_{j=1}^{m} a_{ij}x'_j \geq 2$. De akkor

$$\tilde{z}(\mathbf{x}') = \sum_{j=1}^{m} c'_j x'_j = \sum_{j=1}^{m} (c_j + Rt_j)x'_j = \sum_{j=1}^{m} c_j x'_j + R\sum_{j=1}^{m}\sum_{i=1}^{n} a_{ij}x'_j =$$

$$\sum_{j=1}^{m} c_j x'_j + R\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x'_j \geq \sum_{j=1}^{m} c_j x'_j + R(n+1) \geq R + Rn > \sum_{j=1}^{m} c_j \bar{x}_j + R\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}\bar{x}_j =$$

$$\sum_{j=1}^{m} c_j \bar{x}_j + R\sum_{j=1}^{m} \bar{x}_j \sum_{i=1}^{n} a_{ij} = \sum_{j=1}^{m} c_j \bar{x}_j + R\sum_{j=1}^{m} t_j \bar{x}_j = \sum_{j=1}^{m} (c_j + Rt_j)\bar{x}_j = \tilde{z}(\bar{\mathbf{x}}).$$

A fentiekből következik, hogy (13.3) optimális megoldása eleme az L halmaznak. Másrészt egyszerűen belátható, hogy tetszőleges $\bar{\mathbf{x}} \in L$-re $\tilde{z}(\bar{\mathbf{x}}) = z(\bar{\mathbf{x}}) + Rn$ teljesül. Ebből viszont következik, hogy (13.3) optimális megoldása egyben optimális megoldása (13.2)-nek is és fordítva, amivel a tételt igazoltuk.

Az előző állítást alkalmazhatjuk annak eldöntésére, hogy létezik-e a (13.2) feladatnak lehetséges megoldása. Nevezetesen, a megadott $P_1, \ldots, P_m$ részhalmazokhoz hozzávesszük a $P_{m+1} = \{1, \ldots, n\}$ részhalmazt is egy alkalmasan nagy súllyal. (Például $1 + \sum_{j=1}^{m} c_j$ elegendően nagy súly.) Így teljesül a tétel feltétele. Egyszerűen belátható, hogy megoldva a kibővített (13.2)-höz konstruált (13.3) feladatot, az eredeti (13.2)-nek akkor és csak akkor létezik optimális megoldása, ha a kibővített (13.2) optimuma kisebb, mint $1+\sum_{j=1}^{m} c_j$, azaz P_{m+1} nem szerepel a kibővített feladat optimális megoldásában.

A továbbiakban a (13.1) feladat megoldására szorítkozunk. A problémáról igazolást nyert, hogy az NP-teljes (ld. [2], [107]) feladatok osztályához tartozik. Ennek következtében megoldására különböző speciális esetekre léteznek hatékony, gyors eljárások. Az általános esetre Branch and Bound technikák, valamint heurisztikák kerültek kidolgozásra. Mi most egy $B\&B$ eljárást fogunk felépíteni. Mielőtt erre rátérnénk megállapítjuk a (13.1) feladat bizonyos tulajdonságait, amelyek szükségesek a felépítendő eljáráshoz.

Redukciók

Bizonyos feltételek teljesülése esetén a megoldandó feladat mérete csökkenthető. Az erre vonatkozó, úgynevezett redukciós szabályok több, egymástól független munkában megtalálhatók, többek között a [12], [69] és [125], dolgozatokban. A szabályok megadásához jelölje az $\mathbf{A}$ együtthatómátrix i-edik sorvektorát $\mathbf{a}^{(i)}$ $(i = 1, \ldots, n)$ és j-edik oszlopvektorát $\mathbf{a}_j$ $(j = 1, \ldots, m)$.

1. Ha $\mathbf{a}^{(i)} = \mathbf{e}_k$ ($\mathbf{e}_k$ a k-adik egységvektort jelöli), akkor bármely $\bar{\mathbf{x}} \in L$ lehetséges megoldásra $\bar{x}_k = 1$. De akkor a k-adik oszlop elhagyható, és minden $t \in P_k$-ra a t-edik feltétel is teljesül, így ezen feltételek is elhagyhatók.

2. Ha $\mathbf{a}^{(r)} \geq \mathbf{a}^{(s)}$ valamely $r, s \in \{1, \ldots, n\}$ indexekre, akkor tetszőleges $\bar{\mathbf{x}} \in L$ lehetséges megoldásra $\mathbf{a}^{(s)}\bar{\mathbf{x}} \geq 1$ fennállásából $\mathbf{a}^{(r)}\bar{\mathbf{x}} \geq 1$ fennállása következik. Viszont így az $\mathbf{a}^{(r)}\mathbf{x} \geq 1$ feltétel elhagyható.

3. Ha létezik az $\{1, \ldots, m\}$ halmaznak egy olyan J részhalmaza, hogy $\sum_{t \in J} \mathbf{a}_t \geq \mathbf{a}_k$ és $\sum_{t \in J} c_t \leq c_k$ teljesül valamely $k \in \{1, \ldots, m\}$ indexre, akkor van a feladatnak olyan $\bar{\mathbf{x}}$ optimális megoldása, amelyre $\bar{x}_k = 0$. De akkor elegendő csak ilyen lehetséges megoldások vizsgálatára szorítkozni, azaz $\mathbf{A}$ k-adik oszlopa elhagyható.

Redundáns és prím fedőrendszerek

Legyen adott egy $\mathcal{P} = \{P_{j_1}, \ldots, P_{j_k}\}$ fedőrendszer. A $\mathcal{P}$ fedőrendszer P_{j_s} elemét *redundáns elemnek* nevezzük, ha $\mathcal{P} \setminus \{P_{j_s}\}$ is fedőrendszer. Azt mondjuk, hogy $\mathcal{P}$ *redundáns fedőrendszer*, ha van redundáns eleme. Ellenkező esetben $\mathcal{P}$-t *prím fedőrendszernek* nevezzük. A bevezetett fogalmakkal kapcsolatban nyilvánvalóak a következők.

(i) Valamely redundáns fedőrendszerből elhagyva egy redundáns elemet, olyan fedőrendszerhez jutunk, amelynek súlya kisebb, mint a kiinduló fedőrendszeré.

(ii) A (13.1) feladat bármely optimális megoldása egy prím fedőrendszert ad.

(iii) A $\mathcal{P} = \{P_{j_1}, \ldots, P_{j_k}\}$ fedőrendszer P_{j_s} eleme akkor és csak akkor redundáns, ha bármely $i \in P_{j_s}$-re $\sum_{t=1}^{k} a_{ij_t} \geq 2$ teljesül.

Vegyük észre, hogy (*iii*) alapján tetszőleges fedőrendszerről eldönthető, hogy redundáns-e. Amennyiben redundáns, úgy rendre elhagyva a redundáns elemeket, egy prím fedőrendszert tudunk előállítani. Az előálló prím fedőrendszer függhet a redundáns elemek elhagyásának sorrendjétől. Az elemek elhagyási stratégiájának rögzítésével az előálló prím fedőrendszer egyértelművé tehető. A továbbiakban azonosítjuk $\bar{x}$-t az általa meghatározott fedőrendszerrel, és *prím*($\bar{\mathbf{x}}$)-sal fogjuk jelölni az $\bar{\mathbf{x}}$-ból előállítható prím fedőrendszert.

Lineáris programozási relaxáció

A (13.1) feladatban az $x_j \in \{0, 1\}$ $(j = 1, \ldots, m)$ feltételcsoportot az $x_j \geq 0$ $(j = 1, \ldots, m)$ feltételekkel helyettesítve, a (13.1) feladat lineáris programozási relaxációját kapjuk. A lineáris programozási relaxáció egy $\tilde{\mathbf{x}}$ optimális megoldását vizsgálva, vegyük észre a következőket.

(*iv*) A tekintett $\tilde{\mathbf{x}}$ optimális megoldás minden $\tilde{x}_j$ komponensére $0 \leq \tilde{x}_j \leq 1$ teljesül. Valóban, $\tilde{x}_j > 1$ esetén az illető komponenst 1-re változtatva,

ismét lehetséges megoldást kapnánk, ugyanakkor a célfüggvényérték pedig csökkenne. Következésképp, az optimális megoldás meghatározását illetően elegendő a $0 \leq x_j \leq 1$ $(j = 1, \ldots, m)$ feltételek mellett tekinteni a lineáris programozási relaxációt.

(v) A lineáris programozási relaxáció egy $\tilde{\mathbf{x}}$ optimális megoldásában minden 0-tól különböző komponens helyébe 1-et írva, egy fedőrendszerhez jutunk. Jelölje ezt $int(\tilde{\mathbf{x}})$. Másrészt ebből képezhető a $prím(int\ (\tilde{\mathbf{x}}))$ prím fedőrendszer a redundáns elemek elhagyásával.

B&B eljárás

Az eljárást az 7. fejezetben történt tárgyalásnak megfelelően építjük fel.

I. Az Ω halmaz meghatározása

A (13.1) feladat esetében az Ω halmazt az összes lehetséges bináris m-esek halmazaként definiáljuk, azaz

$$\Omega = \{(x_1, \ldots, x_m) : x_j \in \{0, 1\}, j = 1, \ldots, m\}.$$

Nyilvánvalóan $L \subseteq \Omega$. Másrészt $|\Omega| = 2^m$, így Ω véges.

II. A φ szétválasztási függvény és a g korlátozó függvény megadása

1. Korlátozó függvény

A g korlátozó függvényt Ω speciális részhalmazaira fogjuk definiálni. Ehhez legyen $I \subseteq \{1, \ldots, m\}$, $J \subseteq \{1, \ldots, m\}$, $I \cap J = \emptyset$. Akkor

$$\Omega_{I,J} = \{\mathbf{x} : \mathbf{x} \in \Omega \ \&\ x_i = 1, \ \text{ha}\ i \in I \ \&\ x_j = 0, \ \text{ha}\ j \in J\}.$$

Vegyük észre, hogy $\Omega_{I,J}$ definiálásakor a változók egy részét konstans értéken rögzítettük. A rögzített értékű változókat behelyettesítve a (13.1) feladat feltételrendszerébe, ismételten egy halmazlefedési problémához jutunk. Nevezzük ezt a feladatot (13.1) (I, J)-részproblémájának, és jelölje ezen részprobléma lehetséges megoldásainak halmazát $L_{I,J}$, ahol a lehetséges megoldásokba a rögzített értékű változókat is beleértjük. Egyszerűen belátható, hogy

$$L_{I,J} = L \cap \Omega_{I,J}.$$

De akkor az (I, J)-részprobléma lineáris programozási relaxációjának optimuma alsó korlátja a $z(\mathbf{x})$ ($\mathbf{x} \in L \cap \Omega_{I,J}$) célfüggvényértékeknek, így ezt rendelve az $\Omega_{I,J}$ halmazhoz, a korlátozó függvényünk rendelkezik a kívánt tulajdonságokkal.

A korlátozó függvénnyel kapcsolatban megjegyezzük, hogy amennyiben az (I, J)-részprobléma lineáris programozási relaxációjának $\tilde{\mathbf{x}}$ optimális megoldása egész, úgy rögtön egy fedőrendszerhez jutunk. Ellenkező esetben a $prím(int(\tilde{\mathbf{x}}))$ fedőrendszer képzésével nyerhetünk egy lehetséges megoldást. Így minden egyes korlát meghatározásánál előállítható egy lehetséges megoldás. Egy további fontos észrevétel, hogy amennyiben $\mathbf{c}$ egész és a korlát nem egész, akkor vehetjük a korlát helyett a nála nagyobb egészek közül a legkisebbet, ez még mindig alsó korlát lesz.

2. Szétválasztási fügvény

Legyen $\Omega_{I,J}$ a fentiekben definiált tetszőleges halmaz, ahol $I, J \subseteq \{1, \ldots, m\}$, és $I \cap J = \emptyset$. Nyilvánvaló, hogy $|\Omega_{I,J}| > 1$ akkor és csak akkor teljesül, ha $|I| + |J| < m$. Tegyük fel, hogy az utóbbi reláció fennáll. Akkor van olyan $k \in \{1, \ldots, m\}$, hogy $k \notin I \cup J$. Legyen $I_1 = I \cup \{k\}$, $J_1 = J$, $I_2 = I$, $J_2 = J \cup \{k\}$. Egyszerűen belátható, hogy $\{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}$ az $\Omega_{I,J}$ halmaz egy valódi osztályozása. Most legyen

$$\varphi(\Omega_{I,J}) = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}.$$

A fentiekben formalizált szétválasztás során egy további változónak, x_k-nak az értékét rögzítjük 0 és 1 értéken, és ennek megfelelően bontjuk fel két osztályra a tekintett $\Omega_{I,J}$ halmazt.

A szétválasztási függvény definiálásához egy további pontosítás szükséges. Nevezetesen meg kell adnunk k kiválasztási stratégiáját. Ezt rögzítsük oly módon, hogy a lehetséges változók közül vegyük azt, amelyhez maximális értékű célfüggvényegyüttható tartozik, és több ilyen index esetén ezek közül válasszuk a minimális indexűt.

III. Faépítési stratégia

Alkalmazzuk ismételten a minimális korláttal rendelkező szögpont kiválasztásának stratégiáját.

Ahhoz, hogy teljessé tegyük a $B\&B$ eljárást, meg kell adnunk egy heurisztikát egy induló megoldás meghatározásához. Az egyszerűség kedvéért vegyük azt a heurisztikát, amely nagyság szerint növekvő sorrendbe rendezi a célfüggvényegyütthatókat, és a rendezett sorozatnak megfelelően kapnak 1 értéket a változók addig, amíg elérjük, hogy minden feltétel teljesül.

A redukciók és az eljárás demonstrálására tekintsük a következő feladatot.

13.1. példa

Legyen $I = \{1, \ldots, 7\}$ és

$P_1 = \{1,3,5\}$, $c_1 = 6$, $P_7 = \{1,2,3,4\}$, $c_7 = 5$,
$P_2 = \{2,3,4,7\}$, $c_2 = 4$, $P_8 = \{1,3,7\}$, $c_8 = 2$,
$P_3 = \{3,5,6\}$, $c_3 = 6$, $P_9 = \{2,6\}$, $c_9 = 3$,
$P_4 = \{1,2,4,6\}$, $c_4 = 3$, $P_{10} = \{3,7\}$, $c_{10} = 4$,
$P_5 = \{1,6,7\}$, $c_5 = 3$, $P_{11} = \{5,6\}$, $c_{11} = 3$,
$P_6 = \{3,6,7\}$, $c_6 = 4$.

A következő táblázatban megadtuk a feladat együtthatóit, valamint sorszámozva az alkalmazott redukciós lépéseket.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
$\mathbf{a}^{(1)}$	1	0	0	1	1	0	1	1	0	0	0
$\mathbf{a}^{(2)}$	0	1	0	1	0	0	1	0	1	0	0
$\mathbf{a}^{(3)}$	1	1	1	0	0	1	1	1	0	1	0
$\mathbf{a}^{(4)}$	0	1	0	1	0	0	1	0	0	0	0
$\mathbf{a}^{(5)}$	1	0	1	0	0	0	0	0	0	0	1
$\mathbf{a}^{(6)}$	0	0	1	1	1	1	0	0	1	0	1
$\mathbf{a}^{(7)}$	0	1	0	0	1	1	0	1	0	1	0
$\mathbf{c}$	6	4	6	3	3	4	5	2	3	4	3

Az egymást követő redukciós lépésekben a vektorok mindig az aktuális, az elhagyásokkal, törlésekkel előálló együtthatókra vonatkoznak.

(1) $\mathbf{a}^{(2)} \geq \mathbf{a}^{(4)}$, így $\mathbf{a}^{(2)}$ törölhető.

(2) $\mathbf{a}_8 + \mathbf{a}_{11} \geq \mathbf{a}_3$ és $c_8 + c_{11} < c_3$, ezért $x_3 = 0$, azaz a harmadik oszlop elhagyható.

(3) $\mathbf{a}_4 + \mathbf{a}_8 \geq \mathbf{a}_7$ és $c_4 + c_8 \leq c_7$, így $x_7 = 0$, azaz a hetedik oszlop elhagyható.

(4) $\mathbf{a}_8 + \mathbf{a}_{11} \geq \mathbf{a}_1$ és $c_8 + c_{11} < c_1$, de akkor $x_1 = 0$, és így az első oszlop elhagyható.

(5) $\mathbf{a}^{(5)} = \mathbf{e}_9$, amiből $x_{11} = 1$, és így az ötödik, hatodik sorok törölhetők.

(6) $\mathbf{a}_2 \geq \mathbf{a}_{10}$ és $c_2 \leq c_{10}$, így $x_{10} = 0$ és a tizedik oszlop elhagyható.

(7) $\mathbf{a}_4 \geq \mathbf{a}_9$ és $c_4 \leq c_9$, amiből $x_9 = 0$, azaz a kilencedik oszlop elhagyható.

(8) $\mathbf{a}^{(7)} \geq \mathbf{a}^{(3)}$, azaz a hetedik sor törölhető.

(9) $\mathbf{a}_4 \geq \mathbf{a}_5$ és $c_4 \leq c_5$, ezért $x_5 = 0$ és az ötödik oszlop elhagyható.

(10) $\mathbf{a}_8 \geq \mathbf{a}_6$ és $c_8 \leq c_6$, így $x_6 = 0$ és a hatodik oszlop elhagyható.

A redukciók utáni feladat a következő:

$$\begin{array}{r} x_4 + x_8 \geq 1 \\ x_2 \quad + x_8 \geq 1 \\ x_2 + x_4 \quad \geq 1 \\ x_2, x_4, x_8 \in \{0,1\} \\ \hline 4x_2 + 3x_4 + 2x_8 = z \to \min \end{array}$$

Képezve a heurisztikus megoldást: $\mathbf{x}_0 = (0,1,1)$ és $z(\mathbf{x}_0) = 5$. Így $\mathbf{x}^* = \mathbf{x}_0$ és $z^* = 5$.

Következő lépésként a $g(\Omega)$ korlátot kell kiszámítanunk. Véve a fenti feladat lineáris programozási relaxációját, megszorozva minden egyenlőtlenséget -1-gyel, bevezetve a baloldalon az u_1, u_2, u_3 nemnegatív különbségváltozókat, olyan feladatot kapunk, amely duális szimplex algoritmussal megoldható. Az induló és a befejező szimplex táblázatok a következők.

	x_2	x_4	x_8	
u_1	0	-1	-1^*	-1
u_2	-1	0	-1	-1
u_3	-1	-1	0	-1
	4	3	2	0

	u_2	u_3	u_1	
x_8	$-\frac{1}{2}$	$\frac{1}{2}$	$-\frac{1}{2}$	$\frac{1}{2}$
x_2	$-\frac{1}{2}$	$-\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
x_4	$\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$\frac{1}{2}$
	$\frac{3}{2}$	$\frac{5}{2}$	$\frac{1}{2}$	$-\frac{9}{2}$

Az optimális megoldás $\tilde{\mathbf{x}} = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. Így $int(\tilde{\mathbf{x}}) = (1,1,1)$, és rendre a legkisebb indexű redundáns elemet elhagyva, $prím(int(\tilde{\mathbf{x}})) = (0,1,1)$, azaz $\mathbf{x}_0$ adódik ismét. A $\frac{9}{2}$ korlát helyett vehető 5 korlátként, de akkor vége az eljárásnak ($F_0 = \emptyset$).

Következésképp, a kiindulási feladat

optimális megoldása: $\bar{\mathbf{x}} = (0,0,0,1,0,0,0,1,0,0,1)$ és az

optimum értéke $z(\bar{\mathbf{x}}) = 3 + 2 + 3 = 8$.

Az általunk használt egyszerű heurisztikus eljárás egyik hiányossága, hogy a változók értékadásánál nem veszi figyelembe a korábbi értékadásokat. Többek között ezt küszöböli ki az alábbi, V. Chvatal által javasolt heurisztika.

Heurisztikus eljárás (Chvatal [36])

- *Előkészítő rész.* Legyenek az R és W halmazok üresek.
- *Iterációs rész.* Ha $W = I$, akkor vége az eljárásnak. Az algoritmus által szolgáltatott fedőrendszer R. Ellenkező esetben legyen $P_i \notin R$ az a halmaz, amelyre $P_i \setminus W \neq \emptyset$ és a $c(P_i)/|P_i \setminus W|$ érték minimális. Vegyük fel P_i-t az R halmazba, legyen $W = W \cup P_i$ és térjünk rá a következő iterációra.

Az eljárás helyességének igazolása

A fejezet elején feltételeztük, hogy csak olyan feladatokkal foglalkozunk, amelyeknek létezik lehetséges megoldása. Ebből adódik, hogy amennyiben valamelyik iterációs lépésben $W \subset I$, akkor minden $k \in I \setminus W$ elemhez van olyan $P_{i_k} \in \mathcal{P}$, hogy $P_{i_k} \notin R$ és $k \in P_{i_k}$. Ez azt eredményezi, hogy minden iterációs lépésben tudunk legalább egy P_i halmazt választani, és a W elemszáma legalább 1-gyel nő.

13.2. példa. Alkalmazzuk ezt a heurisztikus eljárást a 13.1 példában vizsgált feladatra. Az első lépésben $c_8/|P_8|$ a minimális hányados, így $R = \{P_8\}$, $W = \{1,3,7\}$. A következő iterációs részben $c_4/|P_4 \setminus W|$ lesz a minimális elem, ezért $R = \{P_8, P_4\}$, $W = \{1,2,3,4,6,7\}$. Ismét rátérve a következő iterációra, $c_{11}/|P_{11} \setminus W|$ a minimális, így $R = \{P_8, P_4, P_{11}\}$, $W = \{1,2,3,4,5,6,7\}$. A következő iterációban $W = I$, így azt kapjuk, hogy az algoritmus által adott fedőrendszer $R = \{P_8, P_4, P_{11}\}$, amelyet a következő értékadás ír le: $\bar{x}_4 = \bar{x}_8 = \bar{x}_{11} = 1$, a többi indexre $\bar{x}_j = 0$. Vegyük

észre, hogy pontosan az előzőekben meghatározott optimális megoldáshoz jutottunk.

A fejezetet két szép eredménnyel zárjuk, melyek közül az első a megismert heurisztikára, a második pedig a halmazlefedési feladat bármely polinomkorlátos heurisztikájára vonatkozik.

13.2. tétel ([36]). *A Chvatal-féle heurisztikus eljárás $H(d) = \sum_{i=1}^{d} 1/i$ - approximációs, ahol d a halmazrendszer egy maximális elemszámú tagjának elemszámát jelöli.*

Bizonyítás. Vegyünk egy tetszőleges halmazlefedési problémát. Hajtsuk végre a Chvatal-féle heurisztikus eljárást. Jelölje $P_{r(1)}, \ldots, P_{r(k)}$ az eljárás által kiválasztott halmazokat a kiválasztás sorrendjében. Minden $j = 0, 1, \ldots, k$ esetén legyen W_j az első j kiválasztott halmaz uniója azaz $W_j = \cup_{i=1}^{j} P_{r(i)}$.

Minden $e \in I$ elemre jelölje $j(e)$ azt, hogy hányadik iterációs lépésben választott az eljárás először olyan halmazt, amely tartalmazza e-t, azaz $j(e) = \min\{1 \leq j \leq k \ : \ e \in P_{r(j)}\}$. Mivel az algoritmus valóban egy fedőrendszert választ ki, ezért a fentiekben definiált $j(e)$ érték minden e esetén létezik. Legyen $y(e)$ az a hányados, amelyet az algoritmus a $j(e)$-edik iterációs részben a $P_{r(j(e))}$ halmaz esetén kiszámolt, azaz legyen $y(e) = c_{r(j(e))}/|P_{r(j(e))} \setminus W_{j(e)-1}|$.

Vegyünk egy tetszőleges P_q halmazt a tekintett feladat halmazrendszeréből. Jelölje k' azt, hogy hányadik iterációs részben lett az utolsó elem a P_q halmazból lefedve, azaz legyen $k' = \max\{j(e) \ : \ e \in P_q\}$. Vizsgáljuk a $\sum_{e \in P_q} y(e)$ összeget. Amennyiben egy i számra és a P_q halmaz valamely e elemére $j(e) = i$, akkor ebben az esetben $y(e) = c_{r(i)}/|P_{r(i)} \setminus W_{i-1}|$. Továbbá azok az e egészek, amelyekre $e \in P_q$ és $j(e) = i$ teljesül pontosan a $P_q \cap (W_i \setminus W_{i-1})$ halmaz elemei. Így azt kapjuk, hogy

$$\sum_{e \in P_q} y(e) = \sum_{i=1}^{k'} \frac{c_{r(i)}}{|P_{r(i)} \setminus W_{i-1}|} |P_q \cap (W_i \setminus W_{i-1})|.$$

Jelölje s_i a $|P_q \setminus W_{i-1}|$ értéket. A W_i halmaz definíciója alapján kapjuk, hogy az s_i értékek sorozata monoton csökken. Továbbá a k' érték definíciója alapján adódik, hogy $s_i \neq 0$, amennyiben $i \leq k'$. Vegyük észre, hogy

$$|P_q \cap (W_i \setminus W_{i-1})| = |P_q \setminus W_{i-1}| - |P_q \setminus W_i| = s_i - s_{i+1}.$$

Másrészt az algoritmus, azért választotta a $P_{r(i)}$ halmazt az i-edik iterációs lépésben, mert ezen halmazra a számolt hányados minimális volt, így a hányados nem lehetett nagyobb, mint a P_q halmaz esetén, azaz

$$\frac{c_{r(i)}}{|P_{r(i)} \setminus W_{i-1}|} \leq \frac{c_q}{|P_q \setminus W_{i-1}|} = \frac{c_q}{s_i}.$$

Következésképpen azt kaptuk, hogy

$$\sum_{e \in P_q} y(e) \leq \sum_{i=1}^{k'} \frac{c_q}{s_i}(s_i - s_{i+1}) = c_q \sum_{i=1}^{k'} \frac{s_i - s_{i+1}}{s_i}.$$

Másrészt

$$\frac{s_i - s_{i+1}}{s_i} \leq \frac{1}{s_i} + \frac{1}{s_i - 1} + \ldots \frac{1}{s_{i+1} + 1} = H(s_i) - H(s_{i+1}),$$

így azt kapjuk, hogy

$$\sum_{e \in P_q} y(e) \leq c_q \sum_{i=1}^{k'} (H(s_i) - H(s_{i+1})) \leq c_q H(s_1).$$

Mivel $|s_1| = |P_q| \leq d$, ezért a fentiekből adódik, hogy $\sum_{e \in P_q} y(e) \leq c_q H(d)$.

Legyen a $\bar{\mathcal{P}}$ fedőrendszer a tekintett halmazlefedési feladat egy optimális megoldása és $c(\bar{\mathcal{P}})$ az optimum értéke. Összegezzük a $\bar{\mathcal{P}}$ megoldásban szereplő halmazokra a $\sum y(e)$ értékeket. A fentiek alapján adódik, hogy ez az összeg legfeljebb $c(\bar{\mathcal{P}}) \cdot H(d)$. Másrészt egy fedőrendszerről van szó, ezért minden $e \in I$ elem szerepel valamelyik halmazban, így a halmazokon vett $\sum y(e)$ értékek összege legalább $\sum_{e \in I} y(e)$. Következésképpen

$$\sum_{e \in I} y(e) \leq c(\bar{\mathcal{P}}) \cdot H(d).$$

Másrészt az $y(e)$ érték definíciója alapján

$$\sum_{e \in I} y(e) = \sum_{i=1}^{k} \sum \{y(e) \ : \ j(e) = i\} = \sum_{i=1}^{k} c_{r(i)}.$$

Továbbá, $\sum_{i=1}^{k} c_{r(i)}$ pontosan a heurisztika által kapott fedés költsége, azaz megmutattuk, hogy ez a költség legfeljebb $H(d) \cdot c(\bar{\mathcal{P}})$, amivel a tétel állítását igazoltuk.

A másik eredmény azt tartalmazza, hogy nem várható konstans approximációs hányadosú polinomkorlátos heurisztika a halmazlefedési problémára.

Nevezetesen M. Bellmore és kollegái [19] igazolták, hogy amennyiben $P \neq NP$, akkor nem létezik konstans approximációs hányadossal rendelkező polinomkorlátos heurisztika a halmazlefedési problémára. Ez más szóval azt jelenti, hogy egy ilyen heurisztikával meghatározott lehetséges megoldáson felvett célfüggvényérték akárhányszorosa lehet az optimum értékének. Ez nem mond ellent a 13.2. tételnek, mivel az abban szereplő összeg bármekkora lehet.

14. Kiszolgálási feladatok

A továbbiakban tárgyalásra kerülő anyag "Facility Location Theory" néven alkotja az operációkutatás egy részterületét. Tulajdonképpen ez a részterület tekinthető a mai modern logisztika kiindulási pontjának, a logisztika bizonyos részeihez biztosít elméleti hátteret. Ez a speciális terület nem túlságosan volt ismert Magyarországon, a hazai szakemberek közül senki sem foglalkozott ezzel a viszonylag új témakörrel. Ennek következtében a témának nincs is kialakult magyar terminológiája. A logisztika térhódításával ez a helyzet lényeges változásban van, egyre többen érdeklődnek a logisztika iránt. Többek között ennek a megnövekedett érdeklődésnek szeretnénk támogatást nyújtani azzal a rövid betekintéssel, amit a következő három fejezet szolgáltat.

Az elméletet a következő típusú feladatok vizsgálatára, megoldására kezdeményezték. Adott a síkon bizonyos számú kliens, akiknek meghatározott igénye van egyfajta homogén szolgáltatásra. Helyezzünk el adott számú kiszolgáló egységet a síkban úgy, hogy egyrészt biztosítsák a kliensek igényeinek kiszolgálását, másrészt a teljes kiszolgálás optimális legyen valamilyen szempont szerint.

Az első ilyen jellegű problémát Fermat vetette fel a XVII. században a következő formában: Adott a síkon három, nem egy egyenesen fekvő pont A, B és C. Határozzuk meg a sík azon X pontját, amelyre az $XA + XB + XC$ távolságösszeg minimális. (Az egyszerűbb jelölés érdekében tetszőleges U, V pontpárra az UV szakasz hosszát UV-vel jelöljük.) Fermat nem adott megoldást a problémára, de még ebben az évszázadban megoldotta azt Torricelli. A következőkben ismertetjük Torricelli megoldását.

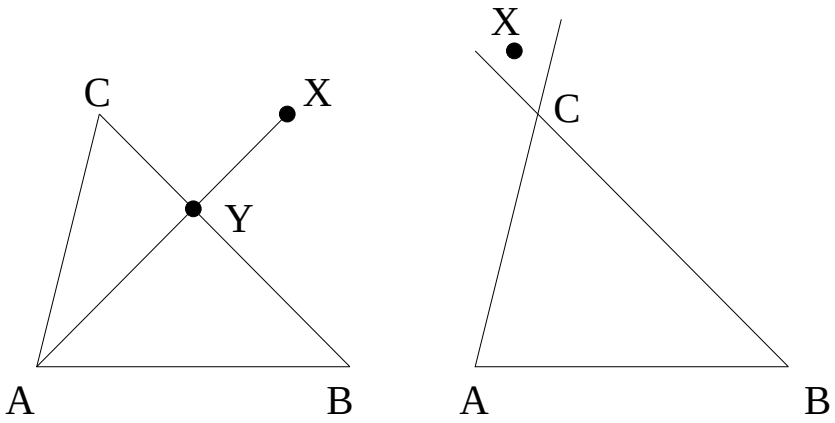
Elsőként vegyük észre, hogy elegendő az X pont keresésénél az $ABC\Delta$ határára és belsejére szorítkozni. Valóban, az $ABC\Delta$-ön kívüli pontok két osztályba sorolhatók, attól függően, hogy

(1) hozzátartoznak a háromszög valamely szögének zárt szögtartományához, vagy

(2) hozzátartoznak egy olyan nyílt szögtartományhoz, amely a háromszög valamely szögének csúcsszöge.

A következő ábra két ilyen pontot tartalmaz. Az (1) esetben X a $CAB\angle$ szögtartomány eleme, amíg a (2) esetben X az $ACB\angle$ csúcsszögének nyílt

tartományába esik.



14.1. ábra. Az (1) és (2) esetek demonstrálása.

Mindkét esetben megmutatjuk, hogy a háromszög határán van olyan Y pont, amelyre $XA + XB + XC > YA + YB + YC$ teljesül.

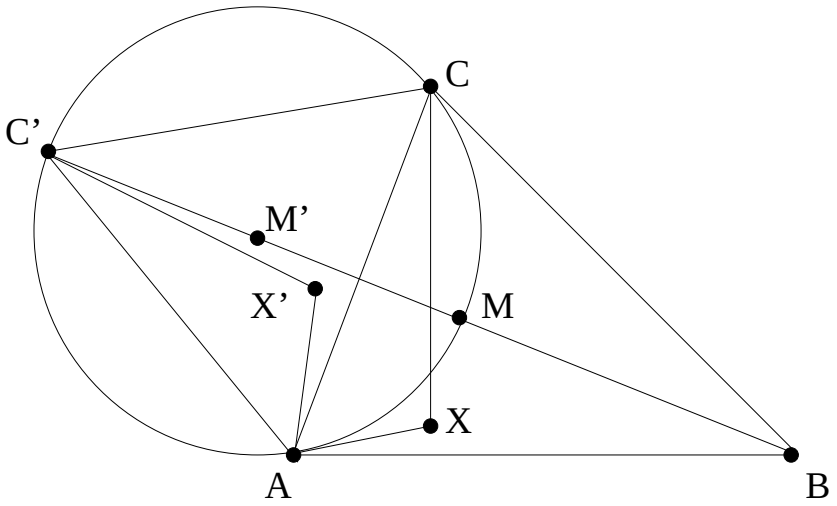
Vegyük észre, hogy az (1) esetben a háromszögegyenlőtlenség miatt $XC + XB > BC = YB + YC$. Másrészt $XA > YA$, és így, $XA + XB + XC > YA + YB + YC$. A (2) esetben az $ABX\Delta$ tartalmazza az $ABC\Delta$-et. De akkor az utóbbi háromszög kerülete kisebb, mint az előzőé, így $XA + XB > CA + CB$. Most a $C = Y$ választással, $XA + XB + XC > YA + YB + YC$, amivel igazoltuk az állítást.

Vizsgáljuk ezek után a háromszög határán és belsejében elhelyezkedő pontokat. Jelöljön X egy ilyen pontot. Forgassuk el A körül az $AXC\Delta$-et 60°-kal. Jelölje rendre C' és X' a C és X pontok képét. Mivel $XA = X'A$ és $XAX'\angle = 60°$, ezért az XAX' háromszög szabályos, így $X'X = XA$. Másrészt $X'C' = XC$. De akkor a vizsgált távolságösszegre $XA + XB + XC = X'X + XB + X'C'$, azaz ez éppen a C' és B pontokat összekötő töröttvonal hossza. Következésképp

$$XA + XB + XC \geq C'B.$$

Most megmutatjuk, hogy az ABC háromszög határán vagy belsejében van olyan M pont, amelyre $AM + BM + CM = C'B$. Ehhez tekintsük ismét az ABC háromszöget és forgassuk el az AC szakaszt A körül 60°-kal. Jelöljük C képét C'-vel és kössük össze a B és C' pontokat. Vegyük fel az ACC' háromszög köré írható körét, és jelölje a kör valamint a BC' szakasz metszéspontját M. Vegyük észre, hogy a $ACM\angle$ és $AC'M\angle$ azonos húrhoz

tartozó kerületi szögek, és mint ilyenek egyenlőek. De akkor elforgatva 60°-kal az AMC háromszöget, az M pont M' képe a BC' szakaszra esik. A háromszöget és az eddigi konstrukciót mutatja a következő ábra.



14.2. ábra. A háromszög elforgatása utáni helyzet.

Az előző gondolatmenettel ismét $AM+BM+CM = M'M+BM+C'M'$, de a kifejezés jobboldala pontosan BC', amivel a kérdéses M pont létezését igazoltuk.

Egy érdekes kérdés, hogy miként jellemezhetjük az M pontot. A forgatás miatt $AC' = AC$, továbbá $C'AC\angle = 60°$, így a $C'AC\Delta$ szabályos. De akkor $AC'C\angle = 60°$, és mivel az $AMCC'$ négyszög húrnégyszög, ezért $AMC\angle = 120°$. Másrészt $ACC'\angle(= 60°)$ és $AMC'\angle$ azonos húrhoz tartozó kerületi szögek, ezért $AMC'\angle = 60°$, amiből $AMB\angle = 120°$. Ezzel azt kapjuk, hogy az M-nél lévő mindhárom szög 120°, azaz M a háromszög azon pontja, amelyből a háromszög mindegyik oldala 120°-os szög alatt látszik. Szokásos a háromszög ezen pontját *izogonális* pontnak nevezni.

A fentiek érvényesek arra az esetre, amikor C' eleme az $ABC\angle$ nyílt szögtartománynak. Egyszerűen belátható, hogy ez akkor és csak akkor teljesül, ha a C-nél és A-nál lévő szögek mindegyike kisebb, mint 120°. Továbbá ismét az AC 60°-os elforgatásával belátható az is, hogy $CAB\angle \geq 120°$ esetében $X = A$-ra lesz minimális a távolságok összege, $ACB\angle \geq 120°$ esetében pedig $X = C$-re lesz minimális a távolságok összege.

A következő kiszolgálási feladatot jóval később, 1909-ben A. Weber [175] fogalmazta meg egy raktárház elhelyezésére, amelynek n számú klienst kell

kiszolgálnia minimális költséggel. Ezt szokásos *Weber-féle modellnek* nevezni. A modell megoldására 1937-ben egy magyar származású matematikus, Vászonyi Endre [176] adott első ízben hatékony eljárást. Vászonyi a második világháború előtt emigrált az USA-ba, ahol a Weiszfeld családnevet használta. Sajnos Vászonyi cikke francia nyelven egy japán újságban jelent meg, így sokan nem tudtak róla. Ráadásul a konvergencia bizonyítása hibás volt. Ezért a 60-as években több szerző ismét publikálta az eljárást.

Az elmélet kifejlesztése alapvetően a 60-as években indult be. Ekkor már a számítástechnika fejlődése miatt lehetővé vált konkrét gyakorlati feladatok megoldása, amely pozitívan visszahatott a téma kutatására, fejlődésére. 1964-ben egy alapvető fogalom került bevezetésre. S. L. Hakimi [85] javasolta, hogy olyan modelleket is kellene vizsgálni, amelyekben a kliensek egy gráf szögpontjaiban, a kiszolgálók pedig a gráfon (beleértve a szögpontokat is) helyezkednek el. Bizonyítást nyert, hogy számos modellben a kiszolgálóknak a gráfon történő elhelyezése visszavezethető a szögpontokba történő elhelyezésükre. Lényegében ez vezetett a Discrete Facility Location Theory kialakulásához.

A vizsgált problémák speciális volta miatt viszonylag hamar elkezdődött egy szeparálódás. Ma már ez egy külön területe az operációkutatásnak és számos, a terület eredményeit összegző munka került publikálásra. Ilyen összegző munkák többek között a [88], [130], [142] és [170] könyvek.

A Facility Location Theory egyes speciális részterületei az alábbiak szerint csoportosíthatók.

FACILITY LOCATION THEORY

I. Folytonos és vegyes modellek

   – determinisztikus modellek
   – sztochasztikus modellek

II. Diszkrét modellek

   – determinisztikus modellek
   – sztochasztikus modellek

A továbbiakban a diszkrét modelleket, ezen belül pedig a determinisztikus modelleket fogjuk vizsgálni. A különböző gyakorlati problémák alapvetően négy alaptípusra vezethetőek vissza, ezek közül fogunk hárommal megismerkedni.

14.1. p-medián probléma

Mielőtt vázolnánk a problémát, felhívjuk a figyelmet arra, hogy a jelen fejezetben olyan hálózatokat fogunk vizsgálni, amelyek gráfjai rendre irányítatlan gráfok. Ezen kívül feltételezzük, hogy a hálózat létező éleihez rendelt súlyok rendre pozitívak. A nemlétező éleknek pedig adjunk 0 súlyt, azaz legyen

$$c'_{ij} = \begin{cases} c_{ij}, & \text{ha } (i,j) \in E, \\ 0 & \text{különben,} \end{cases}$$

ahol E a gráf éleinek a halmazát jelöli és c_{ij} pedig az eredeti súlyokat. Definiálható a gráf tetszőleges két pontja közötti legrövidebb út hossza, és a negyedik fejezetben megismert eljárások alkalmazhatók a legrövidebb utak hosszának meghatározására. Bár a negyedik fejezetben irányított gráfokat tekintettünk, ha minden létező (i,j) élhez felvesszük a (j,i) élet is c_{ij} súllyal, akkor a megismert eljárások, nevezetesen a multiterminális eljárás és a Floyd-Warshall módszer alkalmas a legrövidebb utak meghatározására.

A fenti előkészítés után rátérünk a p-medián probléma definiálására.

p-medián probléma. Legyen adott egy hálózat n csúccsal és jelölje (d_{ij}) a legrövidebb utak hosszainak mátrixát. Minden csúcspontban legyen egy kliens és az i-edik csúcspontban elhelyezkedő kliens igényét jelölje w_i. p számú kiszolgáló (szolgáltató) elhelyezésére van lehetőség, amelyeket a hálózat csúcspontjaiba kell elhelyezni. A szolgáltatók a kiszolgálást a következők szerint végzik:

- Minden szolgáltató korlátlan mennyiségben képes minden klienst kiszolgálni.

- A szolgáltató teljes mértékben kiszolgálja a klienst, tehát nem fordulhat elő, hogy egy kliens igényét két szolgáltató megosztva elégíti ki. (Ez nem lényeges megszorítás, ugyanis megosztott kiszolgálás esetén áttérhetnénk arra a kiszolgálóra, amelyik olcsóbban tudja az illető klienst kiszolgálni.)

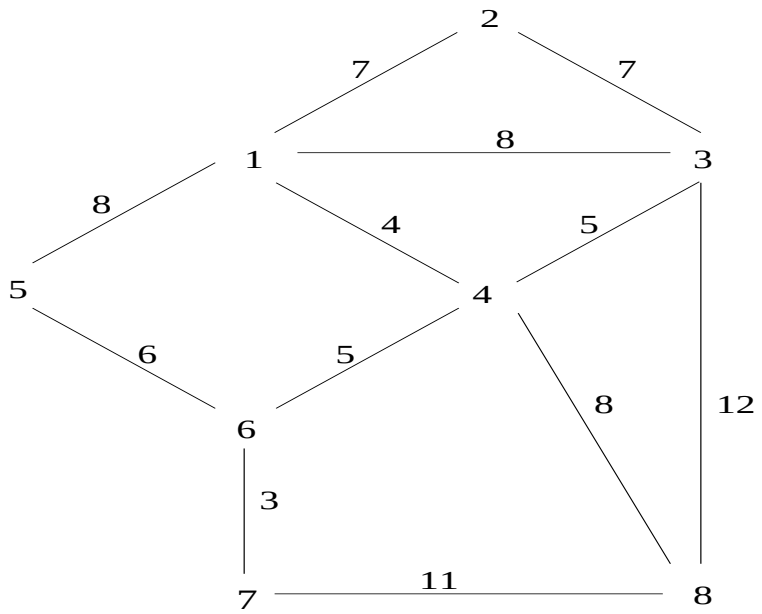
- A kliensek kiszolgálása külön-külön történik, tehát nem megengedett a kliensek megrendeléseinek összevont, egy fuvarral történő megoldása.

Helyezzük el a p számú kiszolgálót úgy, hogy a kliensek kiszolgálásának teljes költsége minimális legyen, ahol az i-edik kliens kiszolgálásának a költsége a j-edik szögpontba telepített kiszolgálóval $w_i d_{ij}$.

Joggal vethető fel az a kérdés, hogy miért nincs megengedve a kiszolgálóknak a gráf éleire történő telepítése. Ezzel kapcsolatban bizonyítást nyert (ld. [85], [88]), hogy amennyiben meg van engedve a kiszolgálóknak az élekre történő telepítése, abban az esetben is van olyan optimális megoldás, hogy minden kiszolgáló a gráf valamely szögpontjában van. Ezen eredmény alapján elegendő az ilyen típusú optimális megoldásokat keresni. Ez viszont azzal biztosítható, hogy a kiszolgálók csak a gráf csúcsaiba helyezhetők el.

Számos gyakorlati probléma felsorolható, amelyek ilyen típusú feladatot eredményeznek. Például képzeljük el, hogy egy külföldi cég, amely valamilyen új árut forgalmaz, raktárakat kíván létrehozni az országban. Ebben az esetben a hálózat lehet az ország úthálózata, a kliensek az egyes városok, mint szögpontok. A cél olyan raktárcsoport kiépítése, amely minimális költséggel biztosítja az áru terítését. Másik példa lehet üzemanyagtároló telepek létesítése. Itt a hálózat ismételten az úthálózat, a kliensek pedig a kiszolgáló kutak lehetnének. Sok érdekes alkalmazás található a [88] könyvben.

Vizsgáljunk most egy példát a legegyszerűbb esetre, az 1-medián problémára.



14.1.1. ábra. A 14.1.1. példa hálózata.

14.1.1. példa. Tekintsük a 14.1.1. ábrán megadott hálózatot, és tegyük fel, hogy minden mennyiségi igény 1-gyel egyenlő, azaz $w_i = 1$, $i = 1, \ldots, 8$. Ez azt jelenti, hogy az i kliens kiszolgálásának költsége megegyezik a legrövidebb út hosszával. (Ez ténylegesen nem jelent megszorítást, mert a mennyiségeket felszorozva a legrövidebb úthosszakkal, áttérhetünk mindig egy másik hálózatra, amelyben már a mennyiségek rendre egyenlőek.) Számítsuk ki minden pontpárra, az őket összekötő legrövidebb úthosszakat. Ezeket az értékeket a következő $\mathbf{D}$ mátrixban adtuk meg.

$$\mathbf{D} = \begin{pmatrix} 0 & 7 & 8 & 4 & 8 & 9 & 12 & 12 \\ 7 & 0 & 7 & 11 & 15 & 16 & 19 & 19 \\ 8 & 7 & 0 & 5 & 16 & 10 & 13 & 12 \\ 4 & 11 & 5 & 0 & 11 & 5 & 8 & 8 \\ 8 & 15 & 16 & 11 & 0 & 6 & 9 & 19 \\ 9 & 16 & 10 & 5 & 6 & 0 & 3 & 13 \\ 12 & 19 & 13 & 8 & 9 & 3 & 0 & 11 \\ 12 & 19 & 12 & 8 & 19 & 13 & 11 & 0 \end{pmatrix}$$

Most vegyük észre, hogy amennyiben az i csúcspontba helyezzük el a kiszolgálót, akkor a kiszolgálás teljes költsége $\sum_{t=1}^{8} d_{ti}$, ami pontosan a $\mathbf{D}$ mátrix i-edik oszlopában szereplő elemek összege. Következésképp, megkapjuk a kiszolgáló optimális (minimális kiszolgálási költséget eredményező) elhelyezését, ha $\mathbf{D}$ minden oszlopára képezzük az elemek összegét, és kiválasztunk egy minimális oszlopösszeget. Ha ez a k-adik oszlophoz tartozik, akkor a kiszolgálót a k csúcsba helyezve kapjuk a minimális kiszolgálási költséget, azaz a k csúcs lesz az optimális megoldás, amit szokásos *1-mediánnak* nevezni. A példában a megfelelő oszlopösszegek rendre, $60, 94, 71, 52, 84, 62, 75, 94$, amelyek közül 52 a minimális, így a kiszolgálót a 4 csúcsba kell elhelyezni.

Bonyolultabbá válik a feladat, ha két kiszolgáló elhelyezésére van lehetőség, azaz egy 2-medián problémát tekintünk. Ebben az esetben feltételezve, hogy a kiszolgálók az i és j csúcsokban vannak elhelyezve, a t csúcsban levő klienst akkor szolgáljuk ki a legkisebb költséggel, ha a hozzá közelebb elhelyezkedő szolgáltató szolgálja ki. A t csúcsban levő kliensnek az i csúcsban levő szolgáltató által történő kiszolgálása esetében a felmerülő költség d_{ti}, amíg a j csúcsból történő kiszolgálás esetén ez a költség d_{tj}. Tehát ilyen elhelyezés mellett a t csúcsban levő kliens minimális kiszolgálási költsége $\min\{d_{ti}, d_{tj}\}$. Ebből adódik a teljes kiszolgálás minimális költsége, ha összegezünk t-re, azaz a tekintett elhelyezés mellett a teljes kiszolgálás minimális költsége $\sum_{t=1}^{n} \min\{d_{ti}, d_{tj}\}$. Ezek után véve az $\{1, \ldots, n\}$ halmaz összes

lehetséges kételemű $\{i, j\}$ részhalmazát, és rendre képezve a fenti összegeket, ezek közül a minimálishoz fog tartozni az optimális elhelyezés. Most az optimális megoldás egy kételemű halmaz lesz, amely megadja, hogy mely csúcsokba kell a kiszolgálókat elhelyezni. Ezt a kételemű halmazt szokásos 2-*medián halmaznak* nevezni.

Kiszámítva a 14.1.1. példára a $\binom{8}{2} = 28$ összeget, azt kapjuk, hogy ez a $\{4, 6\}$ kételemű részhalmazra lesz minimális, azaz a 2-medián halmaz $\{4, 6\}$. Az ezen elhelyezéshez tartozó kiszolgálási költség: 37.

A p-medián problémát többféleképp lehet formalizálni. A következőkben két formalizmust is megadunk, melyekhez szükségesek bizonyos feltételek és jelölések.

Legyen a hálózat gráfja $\mathcal{G} = (V, E)$, ahol

- $V = I \cup J$,
- I pozitív egészek egy nemüres véges halmaza, a kliensek halmaza, akiknek elhelyezkedése ismert és rögzített,
- J pozitív egészek egy nemüres véges halmaza, a kiszolgálók lehetséges helyeinek a pontjai,

továbbá legyen

- minden $i \in I$ és $j \in J$ párra az i kliens kiszolgálásának költsége a j csúcsba telepített kiszolgálónál c_{ij}. Ez azt jelenti, hogy a kiszolgálási költség nem függ a kiszolgálótól, csak a kiszolgálás helyétől.
- p a telepíthető szolgáltatók száma.

Most legyen Q a kiszolgálók elhelyezését rögzítő halmaz, azaz $|Q| = p$ és $Q \subseteq J$. Vizsgáljuk ezen elhelyezés mellett a költségeket. Az $i \in I$ klienst a Q azon helyéről szolgáljuk ki, ahonnét a kiszolgálás költsége minimális, így az i kliens kiszolgálásának a költsége:

$$\min_{j \in Q}\{c_{ij}\}.$$

Másrészt minden klienst ki kell szolgálni, ezért a szolgáltatók rögzített elhelyezése mellett a teljes kiszolgálás költsége:

$$\sum_{i \in I} \min_{j \in Q}\{c_{ij}\}.$$

Következésképp a p-medián problémát leírhatjuk a szokásostól eltérő, alábbi módon:

$$\min_{\substack{Q\subseteq J\\ |Q|=p}} \{\sum_{i\in I} \min_{j\in Q}\{c_{ij}\}\} \tag{14.1.1}$$

Vegyük észre, hogy ez a megfogalmazás általánosítása a korábbi feladatnak, mivel az I és J halmazokra nem tettünk semmiféle kikötést. Nyilvánvalóan $I = J$ esetén az előző feladathoz jutunk.

A (14.1.1) problémához konstruálható egy ekvivalens lineáris programozási feladat a következők szerint.

Minden $j \in J$-re vezessünk be egy x_j bináris változót, és legyen

$$x_j = \begin{cases} 1, & \text{ha a } j \text{ helyre telepítünk szolgáltatót,} \\ 0 & \text{különben.} \end{cases}$$

Továbbá minden $(i, j) \in I \times J$ párra, legyen

$$y_{ij} = \begin{cases} 1, & \text{ha az } i \text{ kilens a } j \text{ helyen levő szolgáltatóval lesz kiszolgálva,} \\ 0 & \text{különben.} \end{cases}$$

Akkor a probléma az alábbi bináris lineáris programozási feladattal írható le:

$$\sum_{j\in J} x_j = p$$

$$y_{ij} - x_j \leq 0, \;\; i \in I, \;\; j \in J$$

$$\sum_{j\in J} y_{ij} = 1, \;\; i \in I \tag{14.1.2}$$

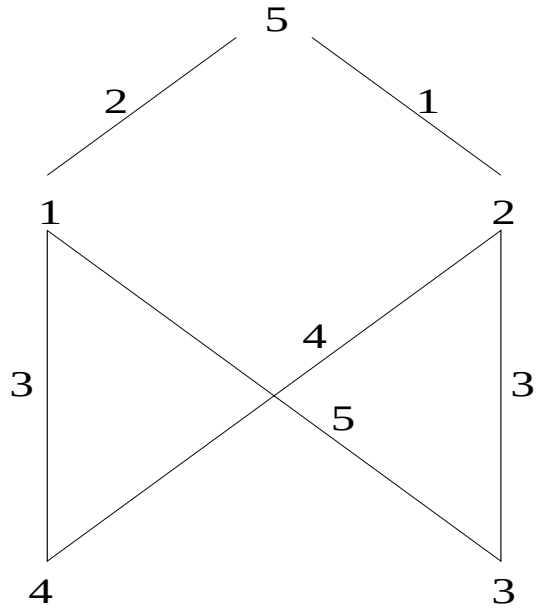
$$x_j \in \{0,1\}, \;\; y_{ij} \in \{0,1\} \;\; i \in I, \;\; j \in J$$

$$\sum_{i\in I, j\in J} c_{ij} y_{ij} = z \to \min$$

A fenti feltételrendszerben az első feltétel azt biztosítja, hogy pontosan p számú szolgáltató kerül telepítésre az előírt helyekre. A második feltételcsoport azt írja elő, hogy csak azokról a helyekről lehet kiszolgálni bármelyik klienst, ahová kiszolgáló lett telepítve. A harmadik feltételcsoport biztosítja, hogy minden kliens kiszolgálásra kerüljön. Végül a célfüggvény a teljes kiszolgálás költségét adja meg.

A fentiek illusztrálására tekintsük a következő példát.

14.1.2. példa. Legyen a kliensek halmaza $I = \{3, 4, 5\}$, a kiszolgálók lehetséges helyeinek halmaza $J = \{1, 2\}$, továbbá legyen $p = 1$. A példa hálózatát adja meg a 14.1.2. ábra.



14.1.2. ábra. A 14.1.2. példa hálózata.

Ekkor a (14.1.1) leírásnak megfelelően keressük a $q = \min_{j \in J}\{\sum_{i=3}^{5} c_{ij}\}$ minimumot. Ekkor $q = \min\{\sum_{i=3}^{5} c_{i1}, \sum_{i=3}^{5} c_{i2}\} = \min\{10, 8\}$. Következésképp az 1-medián pont a 2 csúcs, tehát itt kell elhelyezni a szolgáltatót.

A megfelelő lineáris programozási modellben ((14.1.2) feladat) az x_1, x_2 és az $y_{31}, y_{41}, y_{51}, y_{32}, y_{42}, y_{52}$ változók fognak szerepelni, és a következő feltételrendszert kell kielégíteniük:

$$
\begin{gathered}
x_1 + x_2 = 1 \\
y_{31} - x_1 \leq 0 \\
y_{41} - x_1 \leq 0 \\
y_{51} - x_1 \leq 0 \\
y_{32} - x_2 \leq 0 \\
y_{42} - x_2 \leq 0 \\
y_{52} - x_2 \leq 0 \\
y_{31} + y_{32} = 1 \\
y_{41} + y_{42} = 1 \\
y_{51} + y_{52} = 1 \\
x_j, y_{i,j} \in \{0, 1, \}, \quad i \in \{3, 4, 5\}, j \in \{1, 2\} \\
\hline
\sum c_{ij} y_{ij} = z \to \min
\end{gathered}
$$

A 14.1.2. példa igen szemléletesen mutatja, hogy a viszonylag egyszerűen megoldható feladathoz egy nagy bináris programozási feladatot kapunk, ha áttérünk a megfelelő lineáris programozási feladatra. Ennek a feladatnak a mátrixa 10×8-as lesz. Miután az egészértékű programozási feladatok megoldása igen nehéz feladat, ezért a lineáris programozási feladatnak főleg abban van a jelentősége, hogy relaxációjával korlátot kaphatunk az optimumértékre.

A p-medián problémát illetően igazolást nyert ([68], [105]), hogy a p-medián probléma, ha p-t nem fixáljuk, akkor NP-nehéz. Következésképp nem várható olyan eljárás, amely alkalmas lenne bármilyen p-medián probléma hatékony megoldására. Ennek következtében

- rögzített p mellett kerültek kidolgozásra különböző Branch-and-Bound eljárások, amelyekben a lineáris programozási leírás nagyon jól használható a megfelelő korlátozó függvényhez,
- számos heurisztikus algoritmus került kidolgozásra,
- speciális feladatosztályokat vizsgáltak, és kihasználva ezen osztályok speciális tulajdonságait, megadtak hatékony megoldási technikákat.

A továbbiakban egy ilyen speciális osztállyal és az erre kidolgozott eljárásokkal fogunk megismerkedni.

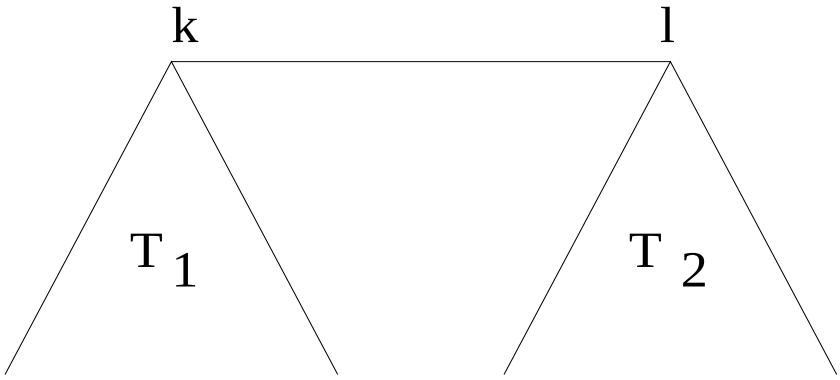
Azzal a speciális esettel foglalkozunk, amelyben a tekintett probléma $\mathcal{G}$ gráfja fa, és erre az esetre megadunk hatékony eljárásokat az 1- és 2-medián problémákra.

További feltételeink:

- minden szögpontban van egy kliens,
- a szolgáltatók bármely szögpontba telepíthetők,
- a klienseknek mennyiségi igénye van, mégpedig az i kliens igénye w_i, amit most *súlynak* fogunk nevezni, azaz w_i lesz az i *csúcs súlya*,
- az i kliens kiszolgálási költsége a j csúcsba telepített szolgáltató révén $w_i d_{ij}$, ahol d_{ij} az i és j csúcsok távolságát jelöli. (Mivel a tekintett gráf fa, így egyetlen út van a két pont között, és d_{ij} ezen út hossza.)

Elsőként az 1-medián problémát fogjuk vizsgálni. Az 1-medián problémában a célfüggvény a gráf csúcsainak a függvénye, konkrétan a j csúcsra megadja a j-be telepített szolgáltató esetén a teljes kiszolgálás költségét. Az adott feltételek mellett ez $z(j) = \sum_{i=1}^{n} w_i d_{ij}$ feltéve, hogy a fa csúcspontjainak halmaza $\{1, \ldots, n\}$.

Legyen most k és l a tekintett $\mathcal{G}$ fa két szomszédos csúcsa, azaz kösse össze őket egy irányítatlan él. Akkor ezt az élet eltávolítva $\mathcal{G}$-ből, $\mathcal{G}$ két diszjunkt részfára esik szét, jelölje ezeket $\mathcal{T}_1$ és $\mathcal{T}_2$. A 14.1.3. ábra mutatja a két részfát.



14.1.3. ábra. A $\mathcal{T}$ fa felbontása két diszjunkt fára.

A kapott részfákra T_1 és T_2-vel fogjuk jelölni az illető részfákhoz tartozó csúcspontok halmazát. Továbbá egy $\mathcal{T}'$ részfához hozzárendelünk egy súlyt, amely a T' halmazban levő csúcsok súlyainak az összege, azaz $w(\mathcal{T}') = \sum_{s \in T'} w_s$. Ezt a súlyt a $\mathcal{T}'$ *részfa súlyának* nevezzük. Ekkor a következő segédtétel érvényes.

14.1.1. segédtétel. $z(k) - z(l) = d_{kl}(w(\mathcal{T}_2) - w(\mathcal{T}_1))$.

Bizonyítás. A definíciók alapján:

$$z(k) = \sum_{i \in T_1} w_i d_{ik} + \sum_{j \in T_2} w_j d_{jk}.$$

$$z(l) = \sum_{i \in T_1} w_i d_{il} + \sum_{j \in T_2} w_j d_{jl} = \sum_{i \in T_1} w_i (d_{ik} + d_{kl}) + \sum_{j \in T_2} w_j (d_{jk} - d_{kl}) =$$

$$\sum_{i \in T_1} w_i d_{ik} + \sum_{j \in T_2} w_j d_{jk} + d_{kl}(w(\mathcal{T}_1) - w(\mathcal{T}_2)) =$$

$$z(k) + d_{kl}(w(\mathcal{T}_1) - w(\mathcal{T}_2)),$$

amivel adódik az állítás.

Használni fogjuk még a következő állítást.

14.1.2. segédtétel. *Ha $w(\mathcal{T}_1) \geq w(\mathcal{T}_2)$, akkor a $\mathcal{T}_1$ fa tartalmaz 1-mediánt.*

Bizonyítás. Az állítást indirekt bizonyítjuk. Ehhez tegyük fel, hogy minden 1-medián pont a $\mathcal{T}_2$ fában van. Ha l 1-medián lenne, akkor a 14.1.1. segédtétel alapján k is 1-medián lenne, ami ellentmondás. Tehát $\mathcal{T}_2$-ben csak l-től különböző 1-mediánok vannak. Akkor ezen 1-mediánok között van egy olyan j csúcs, hogy az l és j csúcsokat összekötő út egyik pontja sem 1-medián. Jelölje ezen út csúcsait $j_1, \ldots, j_k$. Most tekintsük a (j_k, j) él elhagyásával előálló két diszjunkt részfát, és jelölje ezeket $\mathcal{T}$ és $\mathcal{T}'$. Nyilvánvalóan $\mathcal{T}_1 \subseteq \mathcal{T}$ és $\mathcal{T}' \subseteq \mathcal{T}_2$. De akkor

$$w(\mathcal{T}) \geq w(\mathcal{T}_1) \geq w(\mathcal{T}_2) \geq w(\mathcal{T}').$$

A fenti egyenlőtlenségekkel a 14.1.1. segédtétellel adódik, hogy $z(j_k) \leq z(j)$, de akkor j_k 1-medián, ami ellentmondás.

Végül igazoljuk a következő segédtételt.

14.1.3. segédtétel. *Ha $w(\mathcal{T}_1) \geq w(\mathcal{T}_2)$, akkor véve azt a $\mathcal{T}_1'$ fát, amely a $\mathcal{T}_1$ fából a w_k súlynak a $w_k + w(\mathcal{T}_2)$ súllyal történő helyettesítésével áll elő, a $\mathcal{T}_1'$ fa bármely 1-mediánja szintén 1-medián lesz a $\mathcal{G}$ fára vonatkozóan is.*

Bizonyítás. Jelölje a $\mathcal{T}_1'$ fához tartozó 1-medián probléma célfüggvényét z^*. Akkor tetszőleges $m \in T_1$-re $z^*(m) = \sum_{i \in T_1, i \neq k} w_i d_{im} + d_{km}(w_k + w(\mathcal{T}_2))$. Másrészt

$$z(m) = \sum_{i \in T_1} w_i d_{im} + \sum_{j \in T_2} w_j d_{jm} = \sum_{i \in T_1} w_i d_{im} + \sum_{j \in T_2} w_j (d_{jk} + d_{km}) =$$

$$\sum_{i \in T_1} w_i d_{im} + \sum_{j \in T_2} w_j d_{jk} + \sum_{j \in T_2} w_j d_{km} = \sum_{i \in T_1} w_i d_{im} + \sum_{j \in T_2} w_j d_{jk} + d_{km} (\sum_{j \in T_2} w_j) =$$

$$\sum_{i \in T_1, i \neq k} w_i d_{im} + w_k d_{km} + d_{km} w(\mathcal{T}_2) + \sum_{j \in T_2} w_j d_{jk} =$$

$$\sum_{i \in T_1, i \neq k} w_i d_{im} + d_{km}(w_k + w(\mathcal{T}_2)) + \sum_{j \in T_2} w_j d_{jk} = z^*(m) + \sum_{j \in T_2} w_j d_{jk}.$$

Ezek után tegyük fel, hogy $\bar{k}$ 1-medián $\mathcal{T}'$-ben. Ekkor $z^*(\bar{k}) \leq z^*(s)$ teljesül minden $s \in T'$ csúcsra. Most megmutatjuk, hogy $\bar{k}$ 1-medián lesz $\mathcal{G}$-ben is. Ehhez elegendő igazolnunk, hogy a $\mathcal{G}$ fa bármely t elemére $z(\bar{k}) \leq z(t)$ teljesül. A kívánt egyenlőtlenség igazolásához t helyzetétől függően két esetet különböztetünk meg.

1. eset. Tegyük fel, hogy $t \in T_1$. Akkor a z^* és z függvényekre kapott fenti összefüggés alapján

$$z(\bar{k}) = z^*(\bar{k}) + \sum_{j \in T_2} w_j d_{jk},$$
$$z(t) = z^*(t) + \sum_{j \in T_2} w_j d_{jk}.$$

Mivel $z^*(\bar{k}) \leq z^*(t)$, a felírt két egyenlőség alapján adódik a kívánt egyenlőtlenség.

2. eset. Most tegyük fel, hogy $t \in T_2$. Ekkor

$$z(\bar{k}) = z^*(\bar{k}) + \sum_{j \in T_2} w_j d_{jk} \leq z^*(k) + \sum_{j \in T_2} w_j d_{jk} = z(k)$$

mivel $\bar{k}$ 1-medián $\mathcal{T}'$-ben. Ezzel azt kapjuk, hogy $z(\bar{k}) \leq z(k)$. Másrészt a segédtétel feltétele szerint $w(\mathcal{T}_1) \geq w(\mathcal{T}_2)$, amiből a 14.1.1. segédtétellel $z(k) \leq z(l)$ adódik. Most tekintsük az l és t csúcsokat összekötő egyetlen utat, jelölje ezen útnak a végpontoktól különböző csúcsait $u_1, \ldots, u_k$. Akkor rendre elhagyva az $(l, u_1), (u_1, u_2), \ldots (u_k, t)$ éleket és tekintve az elhagyásokkal előálló fapárokat, a 14.1.1. segédtétellel azt kapjuk, hogy

$$z(l) \leq z(u_1),\ z(u_1) \leq z(u_2),\ \ldots\ , z(u_k) \leq z(t).$$

Másrészt $z(\bar{k}) \leq z(k) \leq z(l)$, és így $z(\bar{k}) \leq z(t)$, amivel adódik az állítás.

A bemutatott három segédtételen alapul az alábbi A. J. Goldmantól [77] származó eljárás.

Goldman-féle eljárás ([77])

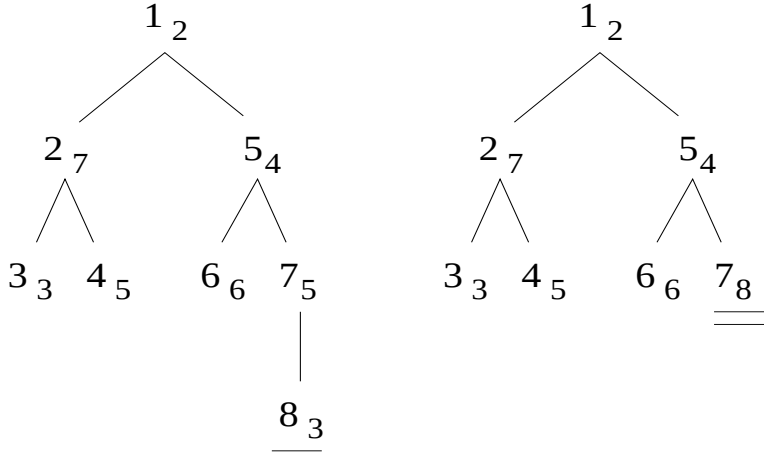
- *1. lépés.* Ha az aktuális $\mathcal{G}$ fa egyetlen szögpontot tartalmaz, akkor vége az eljárásnak, az illető pont a kiindulási feladat 1-mediánja. Ellenkező esetben a 2. lépés következik.

- *2. lépés.* Keressünk az aktuális $\mathcal{G}$ fában egy olyan csúcspontot, amely egyetlen szögponttal szomszédos, azaz a fa levele. Jelöljön i egy ilyen pontot. Ha $w_i \geq w(\mathcal{G})/2$, ahol w_i az aktuális fára vonatkozó súlyt jelöli, akkor vége az eljárásnak; i a kiindulási fában 1-medián. Ellenkező esetben a 3. lépés következik.

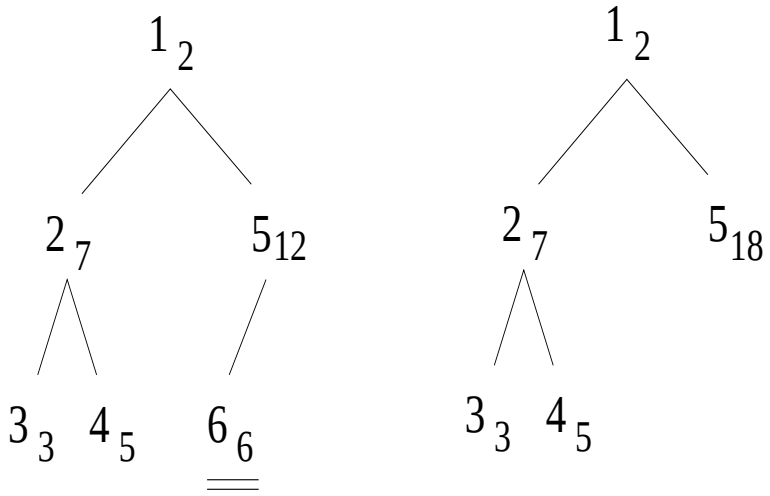
- *3. lépés.* Legyen j a tekintett i csúcs szomszédja az aktuálisa $\mathcal{G}$ fában. Töröljük az aktuális $\mathcal{G}$ fából az i csúcsot valamint az (i, j) élet, továbbá változtassuk meg a törléssel előálló új fa csúcsainak súlyozását úgy, hogy a j csúcs súlyát változtassuk $w_j + w_i$-re. Tekintsük az előállított új fát az új súlyokkal aktuális fának, majd folytassuk az eljárást az 1. lépéssel.

Az eljárás egy végrehajtását illusztrálja az alábbi példa, amelyhez tartozó 14.1.4. és 14.1.5 ábrákon az eljárás során előállításra kerülő fákat adtuk meg a megfelelő súlyokkal. A tekintett fára $w(\mathcal{G}) = 35$.

14.1.3. példa



14.1.4, ábra. A Goldman féle eljárás illusztrálása.



14.1.5. ábra. A Goldman féle eljárás illusztrálása.

Az nyilvánvaló, hogy a vizsgált problémacsoportra ($\mathcal{G}$ fa) mindig létezik 1-medián. Ez azonban nem okvetlen egyértelműen meghatározott. Valóban, ha k 1-medián és szomszédos az l csúccsal, akkor tekintsük ismét a (k,l) él elhagyásával előálló két diszjunkt fát, jelölje ezeket $\mathcal{T}_1$ és $\mathcal{T}_2$. Mivel k 1-medián, ezért $z(k) \leq z(l)$. Továbbá a 14.1.1. segédtétel szerint

$$z(k) - z(l) = d_{kl}(w(\mathcal{T}_2) - w(\mathcal{T}_1)).$$

Így $z(k) = z(l)$ akkor és csak akkor teljesül, ha $w(\mathcal{T}_1) = w(\mathcal{T}_2)$, és ebben az esetben l is 1-medián. Másrészt több 1-medián nem létezhet, ugyanis a pozitív súlyok miatt $z(k') > z(k)$ tetszőleges $k \neq k' \in T_1$-re és $z(l') > z(l)$ tetszőleges $l \neq l' \in T_2$-re. Következésképpen legfeljebb két 1-medián csúcs létezik és ezek szomszédosak. Ennek következtében, ha a Goldman-féle eljárással nyert 1-medián szomszédait megvizsgáljuk, akkor az összes 1-mediánt megkapjuk.

A továbbiakban ugyanazon feltételek mellett a 2-medián problémát fogjuk vizsgálni, és a Goldman-féle eljárás alkalmazásával megadunk egy eljárást a 2-medián halmaz meghatározására. 2-medián esetén a célfüggvény a következő:

$$z(k,l) = \sum_{i=1}^{n} w_i \min\{d_{ik}, d_{il}\}.$$

Most tegyük fel, hogy $\{k,l\}$ egy 2-medián halmaz. Akkor bármely i,j csúcspárra $z(k,l) \leq z(i,j)$ teljesül. Legyen

$$T_1 = \{i : i \in \{1,\ldots,n\} \ \& \ d_{ik} = \min\{d_{ik}, d_{il}\}\}.$$

és $T_2 = V \setminus T_1$. Belátható, hogy felvéve a T_1-beli pontok és a T_2-beli pontok közötti éleket, két diszjunkt részfát kapunk, amelyeket jelöljön $\mathcal{T}_1$ és $\mathcal{T}_2$.

Megmutatjuk, hogy k 1-medián $\mathcal{T}_1$-ben. Ehhez legyen $k' \in T_1$ egy tetszőleges csúcs. Mivel $\{k,l\}$ egy 2-medián halmaz, ezért $z(k,l) \leq z(k',l)$. Másrészt

$$z(k,l) = \sum_{i \in T_1} w_i d_{ik} + \sum_{j \in T_2} w_j d_{jl},$$

$$z(k',l) = \sum_{i \in V} w_i \min\{d_{ik'}, d_{i,l}\} \leq \sum_{i \in T_1} w_i d_{ik'} + \sum_{j \in T_2} w_j d_{jl}.$$

De akkor $\sum_{i \in T_1} w_i d_{ik} \leq \sum_{i \in T_1} w_i d_{ik'}$, amivel igazoltuk, hogy k 1-medián $\mathcal{T}_1$-ben.

A definíciókból egyszerűen adódik, hogy $l \in T_2$, továbbá az előzőekhez hasonlóan belátható, hogy l 1-medián $\mathcal{T}_2$-ben.

A fentiek alapján minden 2-medián halmaz elemei a megfelelő részfákban 1-mediánok lesznek. Ez azt eredményezi, hogy amennyiben vesszük a $\mathcal{G}$ fa összes $\mathcal{T}_1$, $\mathcal{T}_2$ felbontását, (ezek száma $n-1$), és rendre meghatározzuk ezekre az 1-medián párokat, majd kiválasztunk közülük egy olyan párt, amelyre a 2-medián probléma célfüggvénye minimális értéket vesz fel, akkor egy 2-medián halmazt kapunk.

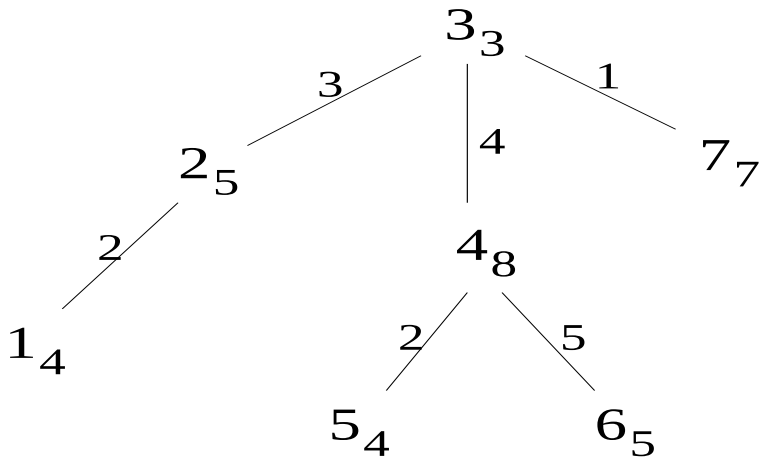
Az eljárást az alábbi példával szemléltetjük.

14.1.4. példa. Tekintsük a 14.1.6. ábrán megadott fát, ahol az élek mentén megadtuk azok hosszát, a csúcsok mellett pedig feltüntettük a csúcsokhoz tartozó súlyokat.

Első lépésként szétbontva a gráfot az $(1,2)$ él elhagyásával, $T_1 = \{1\}$, $T_2 = \{2,\ldots,7\}$, a $\mathcal{T}_1$ fában az 1 csúcs az 1-medián, és alkalmazva a Goldman-féle eljárást a $\mathcal{T}_2$-fában a 4 csúcs adódik 1-mediánként. Ezekre a csúcsokra $z(1,4) = 90$.

A $(2,3)$ él elhagyásával létrejövő részfákban a 2 és 4 csúcsok lesznek a megfelelő 1-mediánok, és $z(2,4) = 78$.

Elhagyva a $(3,4)$ élet, az előálló részfákban a 3 és 4 csúcsok lesznek az 1-mediánok, és $z(3,4) = 75$.



14.1.6. ábra. A 14.1.4. példa súlyozott hálózata.

A $(4,5)$ él elhagyása után az 5 és 3 csúcsok az 1-mediánok, és $z(5,3) = 93$.

A $(4,6)$ élre a 6 és 3 csúcsok adódnak, amelyekre $z(6,3) = 98$.

Végül a $(3,7)$ él elhagyásával a 4 és 7 csúcsok válnak 1-mediánná, és $z(4,7) = 80$.

A kapott célfüggvényértékek közül 75 a minimális, így a $\{3,4\}$ halmaz egy 2-medián halmaz a tekintett feladatban.

A jelen fejezet befejezéseként fontosnak tartjuk megjegyezni, hogy a tárgyalt anyag egy nagyon minimális szelete a témakörnek, inkább csak arra szolgál, hogy bemutassa az alapproblémát. A p-medián problémák jelenleg is a kutatások tárgyát képezik, amit egy új eredménnyel is demonstrálunk. R. E. Burkard professzor, aki az MTA tiszteletbeli tagja, munkatársaival olyan 1-medián problémákat vizsgált, amelyekben meg vannak engedve negatív mennyiségi igények is. A [27] munkában erre a modellre adtak meg egy lineáris idejű megoldó eljárást.

14.2. p-center probléma

A p-center problémát a p-medián feladathoz képest úgy lehet általánosan megfogalmazni, hogy a center problémáknál a klienseknek nincsen mennyiségi igénye, továbbá más a cél, nevezetesen a kiszolgálókat úgy kell elhelyezni, hogy a kiszolgálók és az általuk kiszolgált kliensek távolságának maximuma minimális legyen.

Ilyen jellegű problémákra számos gyakorlati alkalmazást lehet felsorolni, melyek közül itt csak néhányat említünk meg.

Ha egy nagyváros lakosait tekintjük klienseknek és a mentőállomásokat kiszolgálóknak, akkor egy nagyon praktikus cél a mentőállomások egy olyan elhelyezése, hogy minden lakosra, a hozzá legközelebb levő mentőállomás a lehetőségekhez képest a legközelebb legyen.

Szintén egy nagyváros lakosait tekintve klienseknek és a rendőrörsöket kiszolgálóknak, nyilvánvalóan fontos szempont a rendőrörsök olyan elhelyezése, hogy mindenki a lehető legrövidebb idő alatt rendőri segítséghez juthasson.

Egy nagyváros általános iskolai tanulóit tekintve klienseknek és az általános iskolákat kiszolgálóknak, cél lehet az iskolák olyan elhelyezése, hogy minden tanuló a lehető legkevesebbet utazzon a hozzá legközelebb levő iskola eléréséhez.

További példa lehet egy város épületei, mint kliensek, és a tűzoltó állomások, mint kiszolgálók. Ekkor a cél az, hogy bármely épület kigyulladása esetén a legközelelebbi tűzoltó állomásról a lehető leghamarább odaérjenek a tűzoltók.

A felsorolt példákból is nyilvánvaló, hogy a center problémáknál is lehet folytonos és diszkrét problémákat vizsgálni. A továbbiakban csak determinisztikus és diszkrét center problémákkal fogunk foglalkozni.

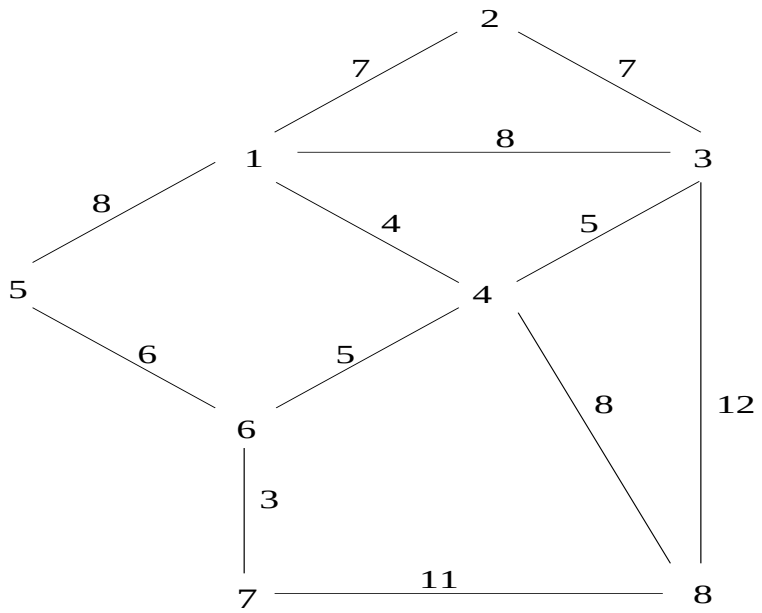
A fentieket pontosítva, a p-center problémát a következőképpen tudjuk definiálni.

p-center probléma. Legyen adott egy n csúcsú irányítatlan gráfon alapuló hálózat a c_{ij} élhosszakkal. Jelölje a legrövidebb utak hosszainak mátrixát (d_{ij}). Továbbá minden csúcspontban legyen egy kliens, és legyen lehetőség p számú kiszolgáló elhelyezésére a hálózat csúcspontjaiba. A kiszolgálást illetően tegyük fel, hogy

- minden kiszolgáló képes kiszolgálni bármelyik klienst,
- minden kiszolgáló kapacitása korlátlan, azaz bármennyi klienst képes kiszolgálni,
- egy klienst csak egy kiszolgáló szolgálhat ki,
- a kliensek kiszolgálása külön-külön történik, azaz nincs összevont szimultán kiszolgálás.

A feladat p számú kiszolgáló elhelyezése a gráf csúcsaiba úgy, hogy a kiszolgálók és az általuk kiszolgált kliensek távolságának maximuma minimális legyen.

Illusztrációként tekintsük a 14.1.1. példát ismét. A tárgyalás könnyebb követhetőségének érdekében megadjuk itt is a hálózatot, valamint a legrövidebb utak **D** mátrixát.



14.2.1. ábra. A 14.1.1. példa hálózata.

$$\mathbf{D} = \begin{pmatrix} 0 & 7 & 8 & 4 & 8 & 9 & 12 & 12 \\ 7 & 0 & 7 & 11 & 15 & 16 & 19 & 19 \\ 8 & 7 & 0 & 5 & 16 & 10 & 13 & 12 \\ 4 & 11 & 5 & 0 & 11 & 5 & 8 & 8 \\ 8 & 15 & 16 & 11 & 0 & 6 & 9 & 19 \\ 9 & 16 & 10 & 5 & 6 & 0 & 3 & 13 \\ 12 & 19 & 13 & 8 & 9 & 3 & 0 & 11 \\ 12 & 19 & 12 & 8 & 19 & 13 & 11 & 0 \end{pmatrix}$$

Vizsgáljuk elsőként a legegyszerűbb esetet, amikor egy kiszolgáló elhelyezésére van lehetőség, azaz oldjuk meg az 1-center problémát. Ha a kiszolgálót a j csúcsba helyezzük, akkor az i kliensnek a kiszolgálótól való távolsága d_{ij}. A d_{ij}, $i = 1, \ldots, 8$ értékek maximuma adja meg a legrosszabb helyzetben levő kliens távolságát a kiszolgálótól. Ez pontosan a $\mathbf{D}$ mátrix j-edik oszlopában levő elemek maximuma. Ezt akarjuk minimalizálni. Ezt megtehetjük úgy, hogy minden j-re képezzük a $\mathbf{D}$ mátrix j-edik oszlopában levő elemek maximumát, majd veszük a kapott maximumok mimimumát. Ha ez a minimum a k-adik oszlopra adódik, akkor a kiszolgálót a k csúcsba kell elhelyezni, és ez esetben k lesz az *1-center* a vizsgált problémában. Képezve az adott $\mathbf{D}$ mátrixra az oszlopokban levő elemek maximumát, rendre a következő értékeket kapjuk: 12,19,16,11,19,16,19,19. Ezek közül 11 a minimális, így a 4 csúcsba kell elhelyezni a kiszolgálót.

A példát interpretálhatjuk úgy, hogy az egy kisebb régió falvainak és az őket összekötő utaknak a hálózata, ahol a távolságok km-ben adottak. A kiszolgáló pedig egy mentőállomás, amelyet el kell helyezni valamelyik faluba. Az eredmény azt adja, hogy amennyiben az állomást a 4 faluba helyezzük el, akkor legfeljebb 11 km távolságra lesz mindenki az állomástól, míg más elhelyezés esetében ez a távolság nagyobb, azaz az ettől eltérő elhelyezés rosszabb kiszolgálást biztosít.

Nehezedik a feladat, ha két kiszolgálót kell elhelyeznünk, azaz 2-center problémát tekintünk. Ha a kiszolgálók az i és j csúcsokba kerülnek elhelyezésre, akkor a t klienst a hozzá közelebb eső kiszolgálóval tudjuk kiszolgálni, mivel a kiszolgálók ki tudnak szolgálni mindenkit és kapacitásuk korlátlan. Ekkor a t csúcsban levő kliens távolsága az őt kiszolgáló szolgáltatótól $\min\{d_{ti}, d_{tj}\}$. A $\min\{d_{ti}, d_{tj}\}$, $t = 1, \ldots, 8$ értékek maximuma a legrosszabb helyzetben levő kliens távolságát adja meg a szolgáltatóktól az adott elhelyezés mellett. Ezt akarjuk minimalizálni, amit megtehetünk úgy, hogy minden $\{i, j\} \subseteq \{1, \ldots, 8\}$ kételemű részhalmazra képezzük a kívánt maximumot, majd a kapott maximumok közül veszünk egy minimálisat. Az

ennek megfelelő kételemű részhalmaz, amit szokásos *2-center halmaznak* nevezni biztosítja a legjobb elhelyezést. A tekintett példára az $\{1, 4\}$ részhalmaz esetében kapjuk a legkisebb maximumot, így a szolgáltatókat az 1 és 4 csúcsokba kell elhelyezni. Ilyen elhelyezés mellett, a legrosszabb helyzetben levő kliens 8 távolságra van az őt kiszolgáló szolgáltatótól.

A fenti interpretáció mellett maradva, most két mentőállomás elhelyezése lehetséges a falvakba, és az 1 és 4 falvakba történő telepítés a legideálisabb. Ekkor a mentőállomásoktól legtávolabb levő falu 8 km távolságra van.

A továbbiakban kétféleképp is formalizáljuk a p-center problémát. Ehhez szükségesek bizonyos feltételek és jelölések.

Legyen a hálózat gráfja $\mathcal{G} = (V, E)$, ahol

- $V = I \cup J$,
- I pozitív egészek egy nemüres véges halmaza, a kliensek halmaza, akiknek elhelyezkedése ismert és rögzített,
- J pozitív egészek egy nemüres véges halmaza, a kiszolgálók lehetséges helyeinek a pontjai,

továbbá legyen

- minden $i \in I$ és $j \in J$ párra az i és j pontokat összekötő legrövidebb út hossza d_{ij},
- p az elhelyezendő szolgáltatók száma.

Most legyen $Q \subseteq J$ és $|Q| = p$. Akkor rendre a Q-beli pontokba elhelyezve a kiszolgálókat, az i kliensnek az őt kiszolgáló szolgáltatótól való távolsága:

$$\min_{j \in Q}\{d_{ij}\}.$$

A legrosszabb helyzetben levő kliens távolsága a kiszolgálójától:

$$\max_{i \in I}\{\min_{j \in Q}\{d_{ij}\}\}.$$

Következésképp a p-center problémát leírhatjuk az alábbi módon:

(14.2.1) $$\min_{\substack{Q\subseteq J\\ |Q|=p}} \{\max_{i\in I}\{\min_{j\in Q}\{d_{ij}\}\}\}$$

Vegyük észre, hogy ez a megfogalmazás általánosítása a korábbi feladatnak, mivel az I és J halmazokra nem írtunk elő semmiféle kikötést. Nyilvánvalóan ha $I = J$, akkor a korábbi feladathoz jutunk.

A medián problémához hasonlóan, a (14.2.1) problémához is konstruálható egy vele ekvivalens lineáris programozási feladat az alábbi módon.

Minden $j \in J$-re legyen

$$x_j = \begin{cases} 1, & \text{ha a } j \text{ helyre telepítünk szolgáltatót,} \\ 0 & \text{különben.} \end{cases}$$

Továbbá minden $(i, j) \in I \times J$ párra, legyen

$$y_{ij} = \begin{cases} 1, & \text{ha az } i \text{ kilens a } j \text{ helyen levő szolgáltatóval lesz kiszolgálva,} \\ 0 & \text{különben.} \end{cases}$$

Akkor a probléma az alábbi bináris lineáris programozási feladattal írható le:

$$\sum_{j\in J} x_j = p$$

$$y_{ij} - x_j \le 0, \;\; i \in I, \;\; j \in J$$

(14.2.2) $$\sum_{j\in J} y_{ij} = 1, \;\; i \in I$$

$$x_j \in \{0,1\}, \;\; y_{ij} \in \{0,1\} \;\; i \in I, \;\; j \in J$$

$$\text{———————————————————-}$$

$$\max_{i\in I,\ j\imath J}\{d_{ij}y_{ij}\} = z \to \min$$

A feltételrendszerben az első feltétel azt biztosítja, hogy pontosan p számú szolgáltató kerül telepítésre az előírt helyekre. A második feltételcsoport garantálja, hogy csak azokról a helyekről lehet kiszolgálni bármelyik klienst, ahová kiszolgáló van telepítve. A harmadik feltételcsoport biztosítja,

hogy minden kliens kiszolgálásra kerüljön. Végül a célfüggvény a legrosszabb helyzetben levő kliens távolságát adja meg az őt kiszolgáló szolgáltatótól.

A p-medián problémára igazolást nyert (ld. [93], [105]), hogy amennyiben p-t nem fixáljuk, akkor ez a probléma NP-nehéz. Ennek következtében nem várható hatékony megoldási eljárás az általános p-center problémára. Így főleg Branch-and-Bound eljárások, valamint heurisztikus eljárások kerültek kidolgozásra, és speciális feladatosztályokra igyekeztek jó eljárásokat megadni. Az utóbbiak közül fogjuk tekinteni a legegyszerűbb speciális osztályt, és erre adunk meg egy eljárást.

Ismét azzal a speciális esettel foglalkozunk, amelyben $\mathcal{G}$ fa. Továbbá feltételezzük, hogy

- minden szögpontban van kliens,
- minden szögpontba lehet kiszolgálót telepíteni.

Az ismertetésre kerülő eljárást G. Y. Handler [86] dolgozta ki. Megadásához szükséges bizonyos jelölések és fogalmak bevezetése. Legyen adott egy $\mathcal{G}$ fa, és jelölje i és j a fa két csúcsát. Akkor pontosan egy, a két pontot összekötő út létezik $\mathcal{G}$-ben, amelyet $[i,j]$-vel, az $[i,j]$ út hosszát pedig d_{ij}-vel jelöljük. Azt mondjuk, hogy az $[i,j]$ *út maximális* $\mathcal{G}$-ben, ha bármely k,l csúcspárra $d_{ij} \geq d_{kl}$ teljesül.

Adott $\mathcal{G}$ fa egy maximális útjának meghatározására ad módszert az alábbi állítás.

14.2.1. segédtétel. *Legyen i a $\mathcal{G} = (V,E)$ fa egy tetszőleges csúcsa, ahol $|V| > 1$. Rendre határozzuk meg az i-ből a fa további csúcsaiba vezető utak hosszait, jelöljön $[i,j]$ egy olyan utat, amelyre ez a hossz maximális. Ugyanezt hajtsuk végre a j csúcsra, és jelöljön $[j,k]$ egy olyan utat, amelyre a hossz maximális. Akkor $[j,k]$ a $\mathcal{G}$ fa egy maximális útja.*

Bizonyítás. Vegyük észre, hogy a j és k csúcsok mindegyike csak egy csúccsal lehet szomszédos, azaz mindkét pont a $\mathcal{G}$ fa levele. Valóban, ellenkező esetben az $[i,j]$ és $[j,k]$ utakat meg lehetne hosszabbítani, ami ellentmondás.

Az állítást indirekt igazoljuk. Ehhez tegyük fel, hogy $[j,k]$ nem maximális útja $\mathcal{G}$-nek. Akkor vannak olyan r,s csúcsok, hogy $d_{rs} > d_{jk}$ és $[r,s]$ maximális út $\mathcal{G}$-ben. Most az i csúcs helyzetétől függően két esetet különböztetünk meg.

1. eset. Tegyük fel, hogy i eleme az $[r, s]$ útnak. Ekkor $i \neq r$, ugyanis $i = r$ esetén azt kapnánk, hogy $d_{rs} = d_{is} \leq d_{ij} \leq d_{jk}$, ami ellentmondás. Teljesen hasonlóan adódik, hogy $i \neq s$. Következésképp, i belső (a végpontoktól különböző) pontja az $[r, s]$ útnak. Ekkor az $[r, s]$ út felbontható az $[r, i]$ és $[i, s]$ utak egyesítésére. Másrészt az $[i, j]$ útnak a kapott két út közül legfeljebb az egyikkel lehet i-től különböző közös pontja, mivel ellenkező esetben kört kapnánk. Tegyük fel, hogy ez $[r, i]$. Akkor j választása miatt $d_{ij} \geq d_{ir}$, továbbá k definíciójával $d_{jk} \geq d_{js}$. Így

$$d_{jk} \geq d_{js} = d_{ji} + d_{is} \geq d_{ir} + d_{is} = d_{rs},$$

ami ellentmondás.

Hasonlóan igazolható a másik eset, valamint az az eset, amikor az $[i, j]$ útnak egyik úttal sincs i-től különböző közös pontja.

2. eset. Tegyük fel, hogy i nem eleme az $[r, s]$ útnak. Ebben az esetben pontosan egy olyan út van a fában, amely i-ből az $[r, s]$ út valamely pontjába vezet. Legyen ez a pont m. Ekkor m nem egyezhet meg az $[r, s]$ út egyik végpontjával sem, ugyanis ilyen egyezés esetén az $[r, s]$ utat meghosszabbíthatnánk, ami ellentmond annak a feltevésünknek, hogy $[r, s]$ maximális út $\mathcal{G}$-ben. Tehát m az $[r, s]$ út egy belső pontja. Tekintsük most azt a részfát, amely az $[r, s]$ és $[i, m]$ utak egyesítéséből áll, jelölje ezt a részfát $\mathcal{T}$. Megmutatjuk, hogy j nem lehet eleme $\mathcal{T}$-nek. Ezt indirekt igazoljuk. Ha $j \in T$, akkor két eset lehetséges, nevezetesen j levele a fának, vagy j-nek két szomszédja van a fában. Elsőként vizsgáljuk azt az esetet, amikor j levele $\mathcal{T}$-nek. Ekkor $j \in \{i, r, s\}$. De $j \neq i$ j definíciója miatt, mivel $|V| > 1$. Tehát $j \in \{r, s\}$. Most $j = r$ esetén, k definíciójával azt kapjuk, hogy $k = s$, ami ellentmondás. Hasonlóan $j = s$ esetén a $k = r$ ellentmondás adódik. Következésképpen j nem lehet levele a $\mathcal{T}$ fának. Ebben az esetben viszont a j csúcsnak két szomszédja van a $\mathcal{G}$ fában, amiből az következik, hogy az $[i, j]$ út meghosszabbítható $\mathcal{G}$-ben, ami ellentmondás. Ezzel igazoltuk, hogy j nem eleme a $\mathcal{T}$ fának. Ebből az következik, hogy pontosan egyetlen út vezet j-ből a $\mathcal{T}$ fa valamely pontjába. Legyen ez a pont u. Az u helyzetétől függően három alesetet különböztetünk meg.

2.1. eset. Tegyük fel, hogy u eleme az $[i, m]$ útnak. Ekkor $d_{ij} \geq d_{ir}$, és így $d_{uj} \geq d_{ur}$. Ebből következik, hogy $d_{js} \geq d_{rs}$. Másrészt k definíciójával $d_{jk} \geq d_{js}$, amivel a $d_{jk} \geq d_{rs}$ ellentmondáshoz jutunk.

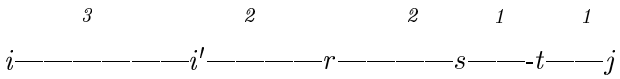
2.2. eset. Tegyük fel, hogy u eleme az $[r, m]$ útnak. Ebben az esetben j definíciójával azt kapjuk, hogy $d_{ij} \geq d_{ir}$, amiből $d_{ju} \geq d_{ru}$ következik. Most $d_{js} = d_{ju} + d_{us} \geq d_{ru} + d_{us} = d_{rs}$, ami k definíciójával a $d_{jk} \geq d_{rs}$ ellentmondáshoz vezet.

2.3. eset. Végül tegyük fel, hogy u eleme az $[m, s]$ útnak. Ekkor $d_{ij} \geq d_{is}$ miatt $d_{ju} \geq d_{su}$. Viszont így $d_{jr} \geq d_{sr}$, de akkor k definíciójával a $d_{jk} \geq d_{rs}$ ellentmondás adódik.

Ezzel a segédtétel bizonyítását befejeztük.

Használni fogjuk a felezéspont következő diszkrét változatát. Legyen $[i, j]$ egy tetszőleges irányítatlan út, amelyben az élhosszak pozitívak. Az $[i, j]$ út r csúcspontját az $[i, j]$ *felezéspontjának* nevezzük, ha az $[i, j]$ út bármely s csúcspontjára $\max\{d_{ri}, d_{rj}\} \leq \max\{d_{si}, d_{sj}\}$ teljesül, ahol d_{ri}, d_{rj}, d_{si}, d_{sj} rendre az $[r, i]$, $[r, j]$, $[s, i]$ és $[s, j]$ utak hosszát jelölik.

Például egy él esetében a tekintett él bármelyik végpontja felezéspontja az élnek. A 14.2.2. ábra egy nemtriviális utat ad meg, amelyre egyszerűen leellenőrizhető, hogy a felezéspontja r.



14.2.2. ábra. Példa felezéspontra.

A diszkrét felezéspont és az 1-centerek kapcsolatát adja meg a következő állítás.

14.2.2. segédtétel. *Legyen a $\mathcal{G}$ fa egy maximális útja $[j, k]$. Akkor a $[j, k]$ út felezéspontja 1-center $\mathcal{G}$-ben.*

Bizonyítás. Ha $[j, k]$ egy él, akkor mivel maximális út $\mathcal{G}$-ben, ezért a j és k csúcsok mindegyikének csak egy szomszédja van, azaz mindkét pont levél. Ekkor viszont a $\mathcal{G}$ fa csak ezt a két pontot és az őket összekötő élet tartalmazza, és így nyilvánvalóan teljesül az állítás. Most tegyük fel, hogy a $[j, k]$ út tartalmaz legalább egy belső (a végpontoktól különböző) pontot. Legyen a $[j, k]$ felezéspontja r. Nyilvánvalóan $r \neq j$ és $r \neq k$.

Ezek után megmutatjuk, hogy r 1-center $\mathcal{G}$-ben. Ehhez azt kell igazolnunk, hogy a $\mathcal{G}$ fa bármely t csúcspontjára

$$q = \max_{i \in V}\{d_{ir}\} \leq \max_{i \in V}\{d_{it}\}$$

teljesül. Az állítást indirekt igazoljuk. Ehhez tegyük fel, hogy a $\mathcal{G}$ fa valamely t csúcspontjára $\max_{i \in V}\{d_{it}\} < q$. Most a t csúcs helyzetétől függően két esetet különböztetünk meg.

1. eset. Elsőként tegyük fel, hogy t a $[j,k]$ út valamely csúcsa. Akkor $\max\{d_{rj}, d_{rk}\} \leq \max\{d_{tj}, d_{tk}\}$, mivel r a $[j,k]$ út felezéspontja. Másrészt mivel a $[j,k]$ út $\mathcal{G}$-nek egy maximális útja, ezért $q = \max\{d_{rj}, d_{rk}\}$. Következésképpen

$$q = \max\{d_{rj}, d_{rk}\} \leq \max\{d_{tj}, d_{tk}\} \leq \max_{i \in V}\{d_{it}\},$$

ami ellentmondás.

2. eset. Most tegyük fel, hogy t nem csúcsa a $[j,k]$ útnak. Ekkor a t csúcsból pontosan egyetlen út vezet $\mathcal{G}$-ben a $[j,k]$ útba. Jelölje ezen út $[j,k]$-ba eső csúcspontját s. Mivel j és k levelei a $\mathcal{G}$ fának, ezért $s \neq j$ és $s \neq k$. Most az r definíciójával azt kapjuk, hogy $\max\{d_{sj}, d_{sk}\} \geq \max\{d_{rj}, d_{rk}\} = q$. Másrészt az élhosszak pozitívitása miatt $\max\{d_{tj}, d_{tk}\} \geq \max\{d_{sj}, d_{sk}\}$, amiből a $\max_{i \in V}\{d_{it}\} \geq q$ ellentmondáshoz jutunk.

Ezzel a 14.2.2. segédtételt igazoltuk.

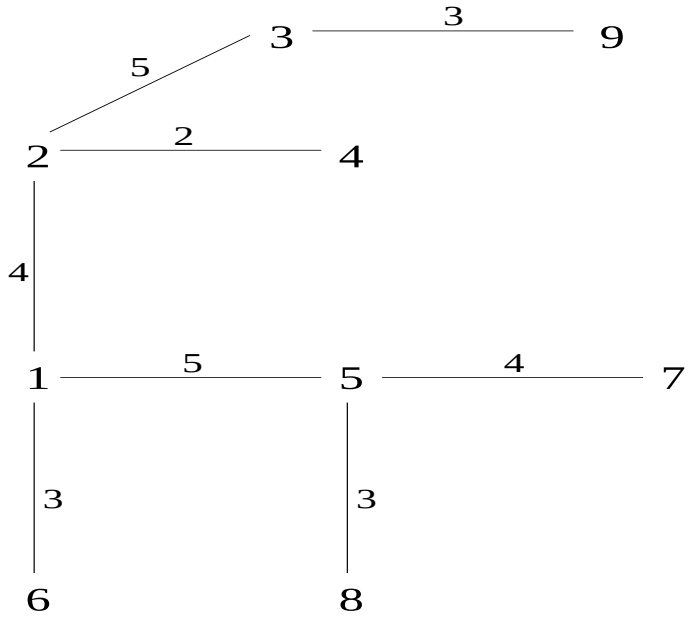
Az előzőek során megadott két segédtételen alapul a következő, G. Y. Handler-től [86] származó eljárás.

Handler eljárása ([86])

- *1. lépés.* Rögzítsünk a $\mathcal{G}$ fában egy i csúcspontot. Határozzunk meg a $\mathcal{G}$ fában az i csúcstól egy legtávolabbi csúcspontot, amelyet jelöljön j.
- *2. lépés.* Határozzunk meg a $\mathcal{G}$ fában a j csúcstól egy legtávolabbi csúcsot, melyet jelöljön k.
- *3. lépés.* Határozzuk meg az 1. és 2. lépésekben kiválasztott j és k pontokat összekötő út felezéspontját. Ezzel az eljárás véget ér; a felezéspont 1-center pont lesz a $\mathcal{G}$ fában.

Az eljárást az alábbi példával illusztráljuk.

14.2.1. példa. Tekintsük a 14.2.3. ábrán adott fát a megadott élhosszakkal. Vegyük kiindulási pontként a 6 csúcsot. Akkor a 6 csúcstól a 9 csúcs van legtávolabb a tekintett fában. Ezek után a 9 csúcstól a 7 csúcs lesz legtávolabb. A $[7,9]$ út felezéspontja 1. Tehát az optimális megoldás az 1 csúcs, azaz ide kell telepíteni a kiszolgálót. Ilyen telepítésnél a kiszolgálótól legtávolabbra levő pont 12 egységnyi távolságra van.



14.2.3. ábra. A 14.2.1. példa hálózata.

Az eljárással kapcsolatban kérdés lehet, hogy miként lehet adott pontból az összes többi pontba vezető utak hosszát gyorsan meghatározni. Ez megtehető úgy, hogy a kitüntetett ponthoz a 0 címkét rendeljük, majd vesszük a pont szomszédait, és ezekhez az ős címkéjében lévő számhoz hozzáadjuk a megfelelő él hosszát, és az így kapott értéket rendeljük címkeként a tekintett ponthoz. Az eljárást tetszőleges címkézett ponttal folytathatjuk, és a nem címkézett szomszédait tudjuk hasonló módon címkével ellátni. Az eljárás akkor fejeződik be, amikor már minden pontnak van címkéje, és ekkor a címkék pontosan a kitüntetett pontból az illető pontokba vezető utak hosszát adják meg.

A fejezet lezárásaként a [142] könyvet követve, bevezetünk egy jelölésrendszert, amellyel könnyen leírhatók a a center problémák különböző változatai, és egyidejűleg utalunk bizonyos eredményekre.

A center problémák egy csoportját egy $X/Y/p/Z$ szimbólumsorozattal fogjuk leírni, ahol

- $X = V$ vagy $X = E$, és X azt jelzi, hogy a kiszolgálókat hová lehet telepíteni. $X = V$ esetén a gráf csúcsaiba, míg $X = E$ esetén a gráf csúcsaiba és éleire is; az utóbbi esetben szokásos *abszolút center problémáról* beszélni,

- $Y = V$ vagy $Y = E$, és Y azt jelzi, hogy hol helyezkedhetnek el a kliensek. $Y = V$ mellett a gráf csúcsaiban, míg $Y = E$ esetén bárhol a gráfon,

- p a szokásos módon a telepíthető szolgáltatók számát rögzíti,

- $Z = N$ vagy $Z = T$, és Z azt mutatja, hogy a tekintett probléma gráfja tetszőleges gráf vagy pedig fa.

A bevezetett jelöléssel, az általunk vizsgált speciális osztályt a $V/V/1/T$ szimbólumsorozat írja le. Fontos megemlíteni, hogy Handler eljárása alkalmazható az abszolút center meghatározására is, annyi eltéréssel, hogy a megfelelő maximális út tényleges (nem diszkrét) felezéspontját kell venni. Azaz a megismert eljárás ezzel a módosítással alkalmas az $E/V/1/T$ problémacsoport feladatainak megoldására is. Handler [87] 1978-ban a korábbi eljárására építve kidolgozott egy lineáris időigényű algoritmust az $E/V/2/T$ problémacsoportra. Az általános abszolút center problémára fák esetén $(E/V/p/T)$ Frederickson és Johnson [61] publikált egy $O(n\log(n))$ műveletigényű algoritmust. Arra az esetre, amikor a gráf nem fa Kariv és Hakimi [105] adott $O(|E|\cdot n)$ műveletigényű eljárást az $E/V/1/N$ problémaosztályra és $O(|E|^p\cdot n^{2p-1}/(p-1)!)$ műveletigényű algoritmust az $E/V/p/N$ feladatosztályra, ahol $p > 1$.

14.3. Kvadratikus hozzárendelési feladat

A jelen fejezetben a kvadratikus hozzárendelési feladattal fogunk megismerkedni, amelyet T. C. Koopmans és M. J. Beckmann [111] vezetett be 1957-ben. A problémát a következőképpen lehet szemléletesen megfogalmazni. Két teljes gráfra épülő hálózatot úgy kell egymásra helyezni, hogy az egymásra helyezett élekhez tartozó súlyok szorzatainak összege minimális legyen. A probléma megoldására egy Branch-and-Bound eljárást építünk fel, melynek korlátozó függvényében speciális hozzárendelési feladatok fognak kulcsszerepet betölteni.

Mielőtt bevezetnénk a modellt, bizonyos előkészületeket teszünk . Legyen adott két azonos elemszámú számsorozat $a_1, \ldots, a_n$ és $b_1, \ldots, b_n$. Minden egyes a_i-hez rendeljük hozzá kölcsönösen egyértelmű módon a másik sorozat egy elemét, amit jelöljön $b_{\varphi(i)}$. Ekkor a hozzárendelést megadó φ leképezés az $\{1, \ldots, n\}$ halmaz egy permutációja. Érdekes kérdés, hogy milyen hozzárendelés (permutáció) mellett lesz minimális az alábbi összeg.

$$\sum_{t=1}^{n} a_t b_{\varphi(t)}. \tag{14.3.1}$$

A kérdésre a megoldást a következő segédtétel szolgáltatja.

14.3.1. segédtétel. ([89]) *A (14.3.1) összeg akkor minimális, ha az* $a_1, \ldots, a_n$ *és* $b_{\varphi(1)}, \ldots, b_{\varphi(n)}$ *sorozatok ellenkezően rendezettek.*

Bizonyítás. Az általánosság megszorítása nélkül feltehetjük, hogy $a_1 \leq \ldots \leq a_n$, ugyanis a tekintett rendezés az elemek átindexezésével elérhető. Most legyen φ az $\{1, \ldots, n\}$ halmaz egy rögzített permutációja, és jelölje b'_i a $b_{\varphi(i)}$ elemet minden $i \in \{1, \ldots, n\}$ indexre. Tegyük fel, hogy $b'_1 \geq \ldots \geq b'_i$ és $b'_i < b'_{i+1}$ valamely $1 \leq i < n$ indexre. Tekintsük azt a $\bar{\varphi}$ permutációt, amelyre $\bar{\varphi}(t) = \varphi(t)$ minden $t \notin \{i, i+1\}$ indexre, és $\bar{\varphi}(i) = \varphi(i+1)$, $\bar{\varphi}(i+1) = \varphi(i)$. Ekkor

$$\sum_{t=1}^{n} a_t b_{\varphi(t)} - \sum_{t=1}^{n} a_t b_{\bar{\varphi}(t)} = \sum_{t=1}^{n} a_t (b_{\varphi(t)} - b_{\bar{\varphi}(t)}) =$$

$$a_i(b'_i - b'_{i+1}) + a_{i+1}(b'_{i+1} - b'_i) = (a_i - a_{i+1})(b'_i - b'_{i+1}).$$

Most vegyük észre, hogy a feltevések miatt az utolsó kifejezés jobboldalán szereplő két tényező közül az első nem pozitív a második pedig negatív, és így a két tényező szorzata nemnegatív. Következésképpen a $\bar{\varphi}$ permutációhoz kisebb vagy egyenlő összeg tartozik, mint a φ permutációhoz, azaz

$$\sum_{t=1}^{n} a_t b_{\bar{\varphi}(t)} \leq \sum_{t=1}^{n} a_t b_{\varphi(t)}.$$

Folytatva a páronkénti elemcseréket, véges sok lépés után olyan φ^* permutációhoz jutunk, amelyre

$$b_{\varphi^*(1)} \geq b_{\varphi^*(2)} \geq \ldots \geq b_{\varphi^*(n)}$$

teljesül, és a φ^*-nak megfelelő (14.3.1) alatti összeg nem nagyobb, mint bármelyik megelőző.

Most véve az $\{1, \ldots, n\}$ halmaznak egy tetszőleges φ permutációját,

- ha φ-vel tágabb értelemben vett monoton csökkenőre rendezhető a $b_1, \ldots, b_n$ sorozat, akkor φ és φ^* csak az azonos értékű elemek felcserélésében térhetnek el egymástól, de ekkor a (14.3.1) összeg nem változik.

- ha φ-vel nem rendezhető tágabb értelemben véve monoton csökkenőre a $b_1, \ldots, b_n$ sorozat, akkor a φ permutációról áttérhetünk egy jobb (kisebb vagy egyenlő (14.3.1) alatti összeg tartozik hozzá) permutációra, amely már monoton csökkenőre rendezi a sorozatot, és az első eset alapján, a kapott permutációhoz tartozó összeg megegyezik a φ^*-hoz tartozó összeggel.

Ezzel a 14.3.1. segédtételt igazoltuk.

14.3.1. Megjegyzés. Teljesen hasonlóan belátható, hogy a (14.3.1) alatti összeg akkor maximális, ha az $a_1, \ldots, a_n$ és $b_{\varphi(1)}, \ldots, b_{\varphi(n)}$ sorozatok azonos rendezettségűek.

A korábbiak során már említettük, hogy a hozzárendelési feladat interpretálható olyan feladatként, amelyben a lehetséges megoldások pontosan az $\{1, \ldots, n\}$ halmaz permutációi, melyek halmazát $\mathcal{P}$-vel jelöltük. Ennek a modellnek valamint a 14.3.1. segédtétel alapján egy hatékony eljárást adunk meg speciális hozzárendelési feladatok megoldására. Ehhez tekintsük a hozzárendelési feladat említett permutációs optimumszámítási modelljét.

$$\varphi \in \mathcal{P} \tag{14.3.2}$$
$$\sum_{i=1}^{n} c_{i\varphi(i)} = z(\varphi) \to \min$$

A hozzárendelési feladat átfogalmazása és a 14.3.1. segédtétel a hozzárendelési feladat egy hatékony megoldási módszerét eredményezi abban a speciális esetben, amikor a c_{ij} célfüggvényegyütthatók felírhatók $c_{ij} = a_i b_j$ alakban valamilyen alkalmas $a_1, \ldots, a_n$ és $b_1, \ldots, b_n$ konstansokkal. Ebben az esetben a (14.3.2) feladat az alábbi feladatra redukálódik:

$$\varphi \in \mathcal{P} \tag{14.3.3}$$
$$\sum_{i=1}^{n} a_i b_{\varphi(i)} = z(\varphi) \to \min$$

A 14.3.1. segédtétel alapján a (14.3.3) feladat célfüggvénye minimális értéket vesz fel, ha az $a_1, \ldots, a_n$ és $b_{\varphi(1)}, \ldots, b_{\varphi(n)}$ sorozatok ellenkezőleg rendezettek. Ebből rögtön adódik a következő megoldó algoritmus.

Eljárás

Rendezzük az $a_1, \ldots, a_n$ sorozat elemeit tágabb értelemben növekedő, a $b_1, \ldots, b_n$ sorozat elemeit tágabb értelemben csökkenő sorrendbe, majd a rendezések szerinti i-edik tagok indexeit feleltessük meg egymásnak minden $i \in \{1, \ldots, n\}$ indexre. Az így képezett φ permutáció optimális megoldása a (14.3.3) feladatnak.

Az eljárás illusztrálására tekintsük a következő példát.

14.3.1. példa. Legyen $a_1 = 1$, $a_2 = 3$, $a_3 = 2$, $a_4 = 1$, $a_5 = 4$, és $b_1 = 3$, $b_2 = 1$, $b_3 = 2$, $b_4 = 3$, $b_5 = 2$. Továbbá legyen a költségmátrix c_{ij} eleme egyenlő $a_i \cdot b_j$-vel minden $i, j \in \{1, \ldots, 5\}$ indexpárra. Ekkor a feladat költségmátrixa a következő, ahol a számítások egyszerűbb követhetőségének érdekében megadtuk a sorok előtt a megfelelő a_i, az oszlopok felett a megfelelő b_j értékeket:

$$\begin{array}{c} \\ 1 \\ 3 \\ 2 \\ 1 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} 3 & 1 & 2 & 3 & 2 \end{array} \\ \begin{pmatrix} 3 & 1 & 2 & 3 & 2 \\ 9 & 3 & 6 & 9 & 6 \\ 6 & 2 & 4 & 6 & 4 \\ 3 & 1 & 2 & 3 & 2 \\ 12 & 4 & 8 & 12 & 8 \end{pmatrix} \end{array}$$

A tekintett példában

$$\begin{array}{ccccccccc} a_1 & \leq & a_4 & \leq & a_3 & \leq & a_2 & \leq & a_5 \\ \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow \\ b_1 & \geq & b_4 & \geq & b_3 & \geq & b_5 & \geq & b_2 \end{array}$$

Ekkor az alábbi permutációt kapjuk:

$$\begin{array}{l} 1 \to 1, \;\; 3 \\ 4 \to 4, \;\; 3 \\ 3 \to 3, \;\; 4 \\ 2 \to 5, \;\; 6 \\ 5 \to 2, \;\; 4 \end{array}$$

A hozzárendelések mellett feltüntettük a megfelelő célfüggvényegyütthatókat, melyek összege 20, így az optimum értéke 20.

A kvadratikus hozzárendelési feladat definiálása előtt a modell néhány egyszerűsített alkalmazását mutatjuk be.

Kórházi részlegek elhelyezése

Adott 5 épületrész, amelyekbe el kell helyezni az alábbi kórházi részlegeket:

- betegfelvétel, adminisztráció,
- sebészet,
- belgyógyászat,
- labor,
- röntgen.

Ismeretes az egyes épületek egymástól való távolsága, továbbá az egyes részlegek közötti betegforgalom nagysága. Helyezzük el úgy a részlegeket, hogy a betegek összes mozgása minimális legyen.

A kórházi részlegek optimális elhelyezésére vonatkozó alkalmazást A. N. Elshafei ismertette az [52] dolgozatban.

Campus kialakítás

Egy egyetem számára adott 6 darab épület, amelyekbe el kell helyezni a következő egységeket:

- tanulmányi osztály,
- előadótermek,
- tanszékek,
- menza,
- sporttermek,
- könyvtár.

Ismeretes az épületek távolsága, valamint az egyes egységek közötti napi hallgatói forgalom. Helyezzük el az adott egységeket az adott épületekbe úgy, hogy az összes hallgatói mozgás minimális legyen.

Az egyetemi egységek elhelyezésére vonatkozó alkalmazást J. W. Dickey és J. W. Hopkins [43] javasolta 1972-ben. A következő alkalmazás kicsit eltér az eddig ismertetett kettőtől. R. E. Burkard és J. Offermann [28] publikálták 1977-ben.

Klaviatúra tervezése

Adott egy üres klaviatúra és ismertek a billentyűk közötti távolságok. Továbbá adott a klaviatúrára felvivendő jelkészlet, és ismert minden jelpárra a szövegekben egymás után történő előfordulásuk gyakorisága. Vigyük fel a billentyűzetre a jeleket úgy, hogy a gyakoriságszor a billentyűtávolságok minden jelpárra képezett összege minimális legyen. (Ez azt eredményezi,

hogy a lehető legkevesebb kézmozgással lehet begépelni a különböző szövegeket.)

Vegyük észre, hogy mindhárom esetben adott egy alaphálózat, amelyben az élhosszak távolságként vannak interpretálva és a hálózat gráfja irányított teljes gráf, továbbá adott egy másik hálózat is, amelyben az élhosszak a csúcsok közötti forgalmat adják meg és ezen hálózat gráfja azonos csúcsszámú irányított teljes gráf. A feladat az , hogy helyezzük a két hálózatot egymásra úgy, hogy az egymást fedő élek élhosszainak szorzatából minden pontpárra képezett összeg minimális legyen. A bevezetett jelölésekkel a problémákat a következő optimumszámítási modellel írhatjuk le.

Kvadratikus hozzárendelési feladat

Legyen adott a $\mathcal{G}_1 = (\{1,\dots,n\}, \{1,\dots,n\} \times \{1,\dots,n\})$ hálózat az (a_{ij}) élhosszakkal, továbbá a $\mathcal{G}_2 = (\{1,\dots,n\}, \{1,\dots,n\} \times \{1,\dots,n\})$ hálózat a (b_{ij}) élhosszakkal. Határozzuk meg az alábbi minimumot:

$$\begin{array}{c} \varphi \in \mathcal{P} \\ \hline \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)} = z(\varphi) \to \min \end{array} \tag{14.3.4}$$

A kvadratikus jelző abból adódik, hogy amennyiben permutációk helyett változókkal írjuk le a hozzárendelést, akkor egy kvadratikus célfüggvényt kapunk. Konkrétan a fenti célfüggvény $a_{ij} b_{\varphi(i)\varphi(j)}$ tagját a

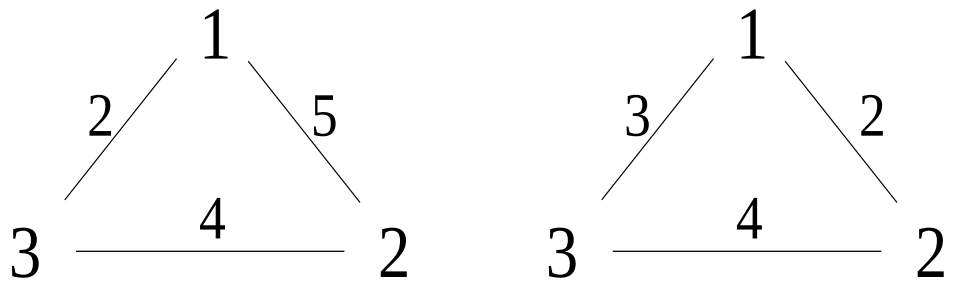
$$\sum_{t=1}^{n}\sum_{r=1}^{n} a_{ij} x_{it} x_{jr} b_{tr}$$

összeggel lehet megadni.

A kvadratikus hozzárendelési feladat demonstrálására tekintsük a következő nagyon egyszerű példát.

14.3.2. példa. Legyen adott a 14.3.1. ábra két hálózata. Tegyük fel, hogy az élhosszak szimmetrikusak és az élhossz mátrixok diagonális elemei rendre 0-val egyenlőek.

A tekintett feladatban a lehetséges megoldások halmaza az $\{1,2,3\}$ halmaz összes permutációja. Ez 6 darab permutáció, így meg tudjuk a feladatot oldani, ha ezen 6 permutációra rendre képezzük a célfüggvényértékeket



14.3.1. ábra. A 14.3.2. példa hálózatai.

és vesszük a legjobbat. A kikötött feltételek a számítások mennyiségének csökkentését célozzák.

A 6 darab permutáció a következő:

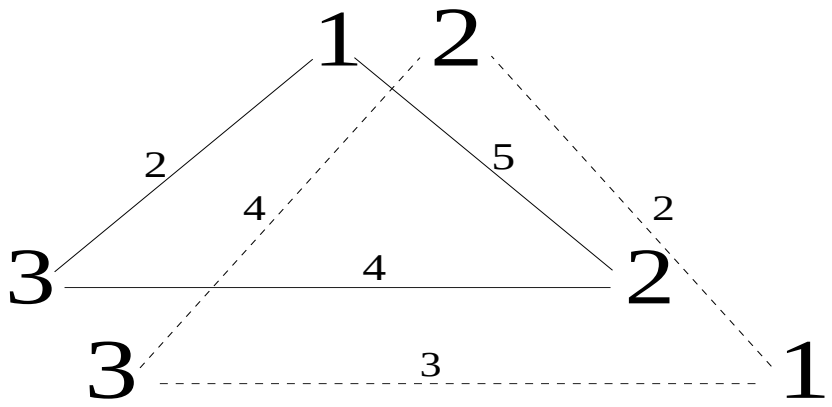
φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
$1 \to 1$	$1 \to 2$	$1 \to 2$	$1 \to 1$	$1 \to 3$	$1 \to 3$
$2 \to 2$	$2 \to 1$	$2 \to 3$	$2 \to 3$	$2 \to 2$	$2 \to 1$
$3 \to 3$	$3 \to 3$	$3 \to 1$	$3 \to 2$	$3 \to 1$	$3 \to 2$

A tekintett permutációkra az alábbi célfüggvényértékek adódnak:

$$z(\varphi_1) = 2(a_{12}b_{12} + a_{13}b_{13} + a_{23}b_{23}) = 2(5 \cdot 2 + 2 \cdot 3 + 4 \cdot 4) = 64$$
$$z(\varphi_2) = 2(a_{12}b_{21} + a_{13}b_{23} + a_{23}b_{13}) = 2(5 \cdot 2 + 2 \cdot 4 + 4 \cdot 3) = 60$$
$$z(\varphi_3) = 2(a_{12}b_{23} + a_{13}b_{21} + a_{23}b_{31}) = 2(5 \cdot 4 + 2 \cdot 2 + 4 \cdot 3) = 72$$
$$z(\varphi_4) = 2(a_{12}b_{13} + a_{13}b_{12} + a_{23}b_{32}) = 2(5 \cdot 3 + 2 \cdot 2 + 4 \cdot 4) = 70$$
$$z(\varphi_5) = 2(a_{12}b_{32} + a_{13}b_{31} + a_{23}b_{21}) = 2(5 \cdot 4 + 2 \cdot 3 + 4 \cdot 2) = 68$$
$$z(\varphi_6) = 2(a_{12}b_{31} + a_{13}b_{32} + a_{23}b_{12}) = 2(5 \cdot 3 + 2 \cdot 4 + 4 \cdot 2) = 62$$

A kapott célfüggvényértékek közül 60 a legkisebb, így a feladat optimális megoldása a φ_2 permutáció. A 14.3.2. ábra mutatja a két hálózat megfelelő egymásrahelyezését.

A kvadratikus hozzárendelési feladat a bonyolult problémák közé tartozik. 1976-ban S. Sahni és T. Gonzalez [157] igazolták, hogy a kvadratikus hozzárendelési feladat NP-nehéz. Ennek következtében Branch-and-Bound eljá- rások kerültek kidolgozásra az optimális megoldások meghatározására, továbbá heurisztikus eljárásokat (ld.. pl. [23]) készítettek szuboptimális megoldások meghatározására. A továbbiakban mi is egy Branch-and-Bound eljárást építünk fel, amelyben a korlátozó függvény meghatározásában a vizsgált speciális hozzárendelési feladatok fontos szerepet töltenek be. Ezt



14.3.2. ábra. A két hálózata egymásrahelyezése.

a korlátozó eljárást P. C. Gilmore [73] és E. L. Lawler [124] egymástól függetlenül javasolták, ezért az irodalomban Gilmore-Lawler korlátozási eljárás néven szokásos említeni.

A korlátozó eljárást csak a teljes feladatra ismertetjük, és csak utalunk rá, hogy a B& B eljárásban milyen eltérések vannak a korlát kiszámításában. A korlát számításának az az alapötlete, hogy a célfüggvényt felbontjuk részösszegekre, melyek közül n számú összeget alulról korlátozunk n^2 speciális hozzárendelési feladat megoldásával, majd a kapott korlátokat felhasználva, a teljes összegre adunk egy alsó korlátot egy általános hozzárendelési feladat megoldásával.

A fentiekben vázoltak megvalósítására tekintsük az $\{1, \ldots, n\}$ halmaz egy tetszőleges φ permutációját. Ekkor

$$z(\varphi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)}.$$

Diagonális elemek kiemelése

$$z(\varphi) = \sum_{i=1}^{n} a_{ii} b_{\varphi(i)\varphi(i)} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}}^{n} a_{ij} b_{\varphi(i)\varphi(j)}$$

Redukció

Minden $j \in \{1, \ldots, n\}$ indexre jelölje α_j az a_{jj} kivételével az $\mathbf{A}$ mátrix j-edik oszlopában levő elemek minimumát, azaz legyen

$$\alpha_j = \min\{a_{tj} : 1 \le t \le n,\ t \ne j\}.$$

Most $\mathbf{A}$ minden oszlopának elemeiből a diagonális elem kivételével vonjuk ki az illető oszlop diagonálistól különböző elemeinek a minimumát, és a kapott mátrixot jelölje $\bar{\mathbf{A}}$. Akkor minden $i \ne j \in \{1, \dots, n\}$ indexpárra

$$a_{ij} = \bar{a}_{ij} + \alpha_j.$$

Hasonlóan β_j-vel jelölve a $\mathbf{B}$ mátrix j-edik oszlopa diagonálistól különböző elemeinek minimumát, és ezt kivonva a j-edik oszlop diagonálistól különböző elemeiből, olyan $\bar{\mathbf{B}}$ mátrixot kapunk, hogy minden $i \ne j \in \{1, \dots, n\}$ indexpárra

$$b_{ij} = \bar{b}_{ij} + \beta_j.$$

Felhasználva a fenti kifejezéseket

$$\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} a_{ij} b_{\varphi(i)\varphi(j)} = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} (\bar{a}_{ij} + \alpha_j)(\bar{b}_{\varphi(i)\varphi(j)} + \beta_{\varphi(j)}) =$$

$$\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} \bar{a}_{ij} \bar{b}_{\varphi(i)\varphi(j)} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} \bar{a}_{ij} \beta_{\varphi(j)} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} \alpha_j \bar{b}_{\varphi(i)\varphi(j)} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} \alpha_j \beta_{\varphi(j)} =$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \bar{a}_{ij} \bar{b}_{\varphi(i)\varphi(j)} + \sum_{j=1}^{n} \beta_{\varphi(j)} \sum_{i=1}^{n} \bar{a}_{ij} + \sum_{j=1}^{n} \alpha_j \sum_{i=1}^{n} \bar{b}_{\varphi(i)\varphi(j)} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \ne i}}^{n} \alpha_j \beta_{\varphi(j)}$$

mivel az összegzés sorrendje felcserélhető és $\bar{a}_{tt} = \bar{b}_{tt} = 0$, $t = 1, \dots, n$. Most vegyük észre, hogy a fenti összeg második tagjában az $\bar{\mathbf{A}}$ mátrix j-edik oszlopában lévő elemek összege szerepel, továbbá a harmadik tagban a $\bar{\mathbf{B}}$ mátrix $\varphi(j)$-edik oszlopában levő elemek összege szerepel. Ezeket az összegeket rendre $\bar{A}_j$-vel és $\bar{B}_{\varphi(j)}$-vel jelölve, a fenti összeg megegyezik az alábbi összeggel:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \bar{a}_{ij} \bar{b}_{\varphi(i)\varphi(j)} + \sum_{j=1}^{n} \bar{A}_j \beta_{\varphi(j)} + \sum_{j=1}^{n} \alpha_j \bar{B}_{\varphi(j)} + (n-1) \sum_{j=1}^{n} \alpha_j \beta_{\varphi(j)} =$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{a}_{ij}\bar{b}_{\varphi(i)\varphi(j)}+\sum_{j=1}^{n}(\bar{A}_j\beta_{\varphi(j)}+\alpha_j\bar{B}_{\varphi(j)}+(n-1)\alpha_j\beta_{\varphi(j)}).$$

Figyelembevéve a diagonális elemek kiemelését is, $z(\varphi)$-re az alábbi összefüggést kapjuk:

$$z(\varphi)=\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{a}_{ij}\bar{b}_{\varphi(i)\varphi(j)}+\sum_{j=1}^{n}(\bar{A}_j\beta_{\varphi(j)}+\alpha_j\bar{B}_{\varphi(j)}+(n-1)\alpha_j\beta_{\varphi(j)}+a_{jj}b_{\varphi(j)\varphi(j)}).$$

Minden $i,j\in\{1,\ldots,n\}$ indexpárra jelölje c_{ij} az

$$\bar{A}_i\beta_j+\alpha_i\bar{B}_j+(n-1)\alpha_i\beta_j+a_{ii}b_{jj}$$

konstanst. Akkor

$$z(\varphi)=\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{a}_{ij}\bar{b}_{\varphi(i)\varphi(j)}+\sum_{j=1}^{n}c_{j\varphi(j)}.$$

Korlátozás

A korlát meghatározásához vizsgáljuk a kapott két szumma közül az elsőt.

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{a}_{ij}\bar{b}_{\varphi(i)\varphi(j)}=\sum_{j=1}^{n}\bar{a}_{1j}\bar{b}_{\varphi(1)\varphi(j)}+\ldots+\sum_{j=1}^{n}\bar{a}_{nj}\bar{b}_{\varphi(n)\varphi(j)}.$$

Most vegyük észre, hogy a fenti egyenlet jobboldalán olyan tagok szerepelnek, amelyek úgy tekinthetők, mint egy speciális hozzárendelési feladat célfüggvényei. A k-adik tag $\sum_{j=1}^{n}\bar{a}_{kj}\bar{b}_{\varphi(k)\varphi(j)}$. Ekkor tekinthető az a $\mathbf{D}$ költségmátrix, amelyet az $\bar{\mathbf{A}}$ mátrix k-adik sorvektora és a $\bar{\mathbf{B}}$ mátrix $\varphi(k)$-adik sorvektora határoznak meg, azaz $d_{ij}=\bar{a}_{ki}\bar{b}_{\varphi(k)j}$. Ennek a hozzárendelési feladatnak az optimuma nem nagyobb, mint $\sum_{j=1}^{n}\bar{a}_{kj}\bar{b}_{\varphi(k)\varphi(j)}$, mivel az utóbbi a célfüggvény értéke egy adott permutációra. Jelölje a tekintett hozzárendelési feladat optimumértékét $f_{k\varphi(k)}$. Akkor

$$=\sum_{j=1}^{n}\bar{a}_{1j}\bar{b}_{\varphi(1)\varphi(j)}+\ldots+\sum_{j=1}^{n}\bar{a}_{nj}\bar{b}_{\varphi(n)\varphi(j)}\geq\sum_{i=1}^{n}f_{i\varphi(i)}$$

és így $z(\varphi) =$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{a}_{ij}\bar{b}_{\varphi(i)\varphi(j)} + \sum_{j=1}^{n} c_{j\varphi(j)} \geq \sum_{j=1}^{n} f_{j\varphi(j)} + \sum_{j=1}^{n} c_{j\varphi(j)} = \sum_{j=1}^{n}(f_{j\varphi(j)} + c_{j\varphi(j)}).$$

Most ismét vegyük észre, hogy a fenti egyenlőtlenség jobboldalán egy olyan hozzárendelési feladat célfüggvényértéke szerepel a φ permutáción, amelynek költségmátrixa $(f_{ij} + c_{ij})$. Másrészt mind a c_{ij}, mind az f_{ij} értékeket ki tudjuk számítani. Valóban,

$$c_{ij} = \bar{A}_i\beta_j + \alpha_i\bar{B}_j + (n-1)\alpha_i\beta_j + a_{ii}b_{jj}$$

továbbá az f_{kl} együttható azon speciális hozzárendelési feladat optimumértéke, amelynek költségmátrixa $(\bar{a}_{ki}\bar{b}_{lj})$. Az (f_{kl}) mátrix meghatározásához n^2 számú speciális hozzárendelési feladatot kell megoldanunk.

Az $(f_{ij} + c_{ij})$ költségmátrixú hozzárendelési feladatot megoldva, az optimum f értéke nem lesz nagyobb, mint a $\sum_{j=1}^{n}(f_{j\varphi(j)} + c_{j\varphi(j)})$ érték mivel az utóbbi a célfüggvény értéke egy rögzített pontban. Következésképpen

$$z(\varphi) \geq f$$

és mivel φ tetszőleges permutáció volt, azt kapjuk, hogy f alsó korlátja a célfüggvényértékeknek a lehetséges megoldások halmazán.

A bemutatott korlátozó eljárással kapcsolatban fontos megjegyezni a következőket, amelyeket majd alkalmazunk a Branch-and-Bound eljárás során felépítendő B&B fa valamely szögpontjához tartozó lehetséges megoldásokon felvett célfüggvényértékek alsó korlátjának meghatározására.

Vegyük észre, hogy tetszőleges φ permutációra

$$z(\varphi) \geq \sum_{j=1}^{n}(f_{j\varphi(j)} + c_{j\varphi(j)})$$

teljesül. A felépítendő Branch-and-Bound eljárásban a permutációkat bináris mátrixokkal fogjuk reprezentálni, és ezen mátrixok bizonyos elemeit 1 értékkel, más elemeit 0 értékkel rögzítjük. A korábbiakhoz hasonlóan I jelöli azon indexpárok halmazát, amelyekhez tartozó változók értéke 1-nek lett rögzítve, és J jelöli azon indexpárok halmazát, amelyekhez tartozó változók 0 értékkel lettek rögzítve. Ez azt jelenti, hogy a B&B fa megfelelő szögpontjához pontosan azok a permutációk (mátrixok) tartoznak, amelyek az I és J halmazok által megadott értékadásokat kielégítik. Másrészt ezen

permutációk (mátrixok) a fenti egyenlőtlenség jobboldalán szereplő hozzárendelési feladatnál leírhatók úgy, hogy vesszük a megfelelő $(A(\mathbf{F}+\mathbf{C}), I, J)$ tiltásos hozzárendelési feladatot. De akkor ezen hozzárendelési feladat optimuma alsó korlátja lesz az illető szögponthoz tartozó lehetséges megoldásokon felvett kvadratikus célfüggvény értékeinek. A korlátozó függvényhez pontosan erre van szükségünk.

A továbbiakban felépítünk egy Branch-and-Bound eljárást a kvadratikus hozzárendelési feladat megoldására. Ehhez tegyük a következő megállapodásokat.

(1) Legyen Ω az $n \times n$-es bináris mátrixok halmaza.

(2) Alkalmazzuk a standard levélkiválasztási stratégiát, azaz minden lépésben válasszunk az élő levelek közül egy olyan levelet, amelynek minimális a korlátja.

(3) A szétválasztást hajtsuk végre a következők szerint. Az eljárás során mindig egy $\Omega_{I,J}$ halmazt fogunk osztályozni valamely x_{rs} szabad $((r,s) \notin I \cup J)$ változója szerint. Az $\Omega_{I,J}$ halmazt két diszjunkt részhalmazra bontjuk fel; az egyikben lesznek azok a mátrixok, amelyekben $x_{rs} = 1$, a másikban pedig azok a mátrixok, amelyekben $x_{rs} = 0$. Ez nyilvánvalóan egy nemtriviális osztályozása $\Omega_{I,J}$-nek. Formálisan:

$$I_1 = I \cup \{(r,s)\}, \qquad J_1 = J, \qquad I_2 = I, \qquad J_2 = J \cup \{(r,s)\}$$

és

$$\psi(\Omega_{I,J}) = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\},$$

ahol most ψ jelöli a szétválasztási függvényt.

A szétválasztási függvénynél fontos, hogy miként választjuk ki az x_{rs} változót a szabad változók közül. Ehhez felhasználjuk az $(A(\mathbf{F}+\mathbf{C}), I, J)$ tiltásos hozzárendelési feladat optimális megoldását meghatározó utolsó költségmátrixot, amit jelöljön $\bar{\mathbf{C}} = (\bar{c}_{ij})$. A $\bar{\mathbf{C}}$ mátrixban az optimális megoldást meghatározó minden olyan 0^*-ra, amelynek indexe nem eleme I-nek, képezzünk egy értéket a következők szerint. Tegyük fel, hogy a tekintett 0^* az i-edik sorban és a j-edik oszlopban van. Ekkor legyen

$$u_i = \min_{\substack{1 \le t \le n \\ t \ne j}} \bar{c}_{it}, \qquad v_j = \min_{\substack{1 \le t \le n \\ t \ne i}} \bar{c}_{tj}, \qquad w_{ij} = u_i + v_j.$$

Nyilvánvaló, hogy $|I| < n$ esetén legalább egy w_{ij} érték létezik. A meghatározott w_{ij} értékek közül válasszunk egy maximálisat, legyen ez w_{rs}. Akkor x_{rs} szabad változó, és hajtsuk végre a szétválasztást x_{rs} szerint.

A szétválasztással kapcsolatban megjegyezzük a következőket.

Az eljárás során előálló I halmazok rendre kölcsönösen egyértelmű leképezéseket írnak le. Ez az $I = \emptyset$ esetre nyilvánvalóan igaz. Most tegyük fel, hogy az eljárás során előálló valamely $1 \leq |I| < n-1$ halmazra is teljesül az állítás, és legyen a fentiek szerint kiválasztott szabad változó x_{rs}. Akkor a megfelelő tiltásos hozzárendelési feladat optimális megoldását meghatározó $(\bar{c}_{ij})$ mátrixban $\bar{c}_{rs} = 0^*$, és a tekintett hozzárendelési feladat optimális megoldása olyan φ permutáció, amely kiterjesztése az I által meghatározott kölcsönösen egyértelmű leképezésnek. Ezen φ permutációnak egy restrikcióját határozza meg az $I \cup \{(r,s)\}$ halmaz, de akkor az $I \cup \{(r,s)\}$ által meghatározott leképezés nyilvánvalóan kölcsönösen egyértelmű.

Előfordulhat, hogy az $x_{rs} = 0$ értékadás azt eredményezi, hogy a feltételeket kielégítő mátrixok r-edik sorának vagy s-edik oszlopának minden elemére a 0 érték lesz előírva. Nyilvánvaló, hogy az ilyen mátrixok egyike sem határoz meg permutációt, ezért $\Omega_{I_2,J_2} \cap L = \emptyset$, ahol a szétválasztás leírásánál használt jelölést alkalmaztuk. Ezt a szituációt az eljárás lekezeli úgy, hogy a megfelelő tiltásos hozzárendelési feladat optimuma M lesz, így az Ω_{I_2,J_2} szögponthoz az eljárás az M korlátot rendeli, ami azt eredményezi, hogy Ω_{I_2,J_2} nem lesz élő levél a következő lépésben.

A fentieknek megfelelően épül fel a következő eljárás.

Eljárás

Előkészítő rész

Vegyük az $(1, 2, \ldots, n)$ ciklikus permutációt, jelölje ezt $\bar{\varphi}$. Legyen $x^* = \bar{\varphi}$ és $\bar{z} = z(\bar{\varphi})$. Határozzuk meg az $A(\mathbf{F} + \mathbf{C})$ hozzárendelési feladat optimális megoldását magyar módszerrel; jelölje a kapott optimális megoldást φ_0. Rendeljük Ω-hoz a kapott optimumértéket. A φ_0 ismeretében aktualizáljuk az x^* és $\bar{z}$ változók értékeit. Legyen $r = 0$ és

$$F_0 = \begin{cases} \{\Omega\}, & \text{ha } g(\Omega) < \bar{z}, \\ \emptyset & \text{különben.} \end{cases}$$

Iterációs rész (r-edik iteráció)

- *1. lépés.* Ha $F_r = \emptyset$, akkor vége az eljárásnak, x^* tartalmazza az optimális megoldást. Ellenkező esetben folytassuk az eljárást a 2. lépéssel.

- *2. lépés.* Válasszunk F_r elemei közül egy olyan halmazt, amelyhez minimális korlát tartozik, jelölje ezt $\Omega_{I,J}$. Alkalmazzuk a ψ szétválasztási függvényt $\Omega_{I,J}$-re. Legyen

$$\psi(\Omega_{I,J}) = \{\Omega_{I_1,J_1}, \Omega_{I_2,J_2}\}.$$

  Ha $|I_1| = n - 1$, akkor határozzuk meg azt az egyetlen permutációt, amely kiterjesztése az I_1 halmaz által meghatározott leképezésnek, majd az előállított permutációval aktualizáljuk az x^* és $\bar{z}$ változók értékeit, továbbá rendeljük az Ω_{I_1,J_1} halmazhoz a permutáción felvett kvadratikus célfüggvényértéket alsó korlátként. Ezt követően folytassuk az eljárást a 4. lépéssel. Ha $|I_1| < n - 1$, akkor folytassuk az eljárást a 3. lépéssel.

- *3. lépés.* Oldjuk meg az $A(\mathbf{F} + \mathbf{C}), I_1, J_1)$ hozzárendelési feladatot. A kapott optimumértéket rendeljük az Ω_{I_1,J_1} halmazhoz és a kapott optimális megoldással aktualizáljuk az x^* és $\bar{z}$ változók értékeit, majd folytassuk az eljárást a 4. lépéssel.

- *4. lépés.* Oldjuk meg az $A(\mathbf{F} + \mathbf{C}), I_2, J_2)$ hozzárendelési feladatot. A kapott optimumértéket rendeljük az Ω_{I_2,J_2} halmazhoz és a kapott optimális megoldással aktualizáljuk az x^* és $\bar{z}$ változók értékeit. Ezek után legyen

$$F_{r+1} = \{\Omega_{I',J'} : \Omega_{I',J'} \in (F_r \setminus \Omega_{IJ}) \cup \psi(\Omega_{I,J}) \ \ \& \ \ g(\Omega_{I',J'}) < \bar{z}\}.$$

  Növeljük r értékét 1-gyel, majd folytassuk az eljárást a következő iterációval.

A fenti eljárással kapcsolatban fontos megjegyezni, hogy az alkalmazott korlátozó függvénynél élesebb korlátozó függvényt is lehet használni. Élesebb korlátozó függvényt kapunk abban az esetben, ha a tiltásokat az **A** és **B** mátrixokba vesszük fel, és minden egyes lépésben alkalmazzuk a Gilmore-Lawler féle korlátozási eljárást ezekre a módosított mátrixokra. Mivel az utóbbi korlátszámítás lényegesen számításigényesebb, ezért alkalmaztuk az egyszerűbb korlátszámítási eljárást.

Az eljárás illusztrálására tekintsünk a következő példát.

14.3.3. példa. Vegyünk két irányított teljes hálózatot rendre három csúccsal, melyek élhosszait az alábbi **A** és **B** mátrixok adják meg.

$$\mathbf{A} = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 4 & 1 \\ 2 & 1 & 3 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} 3 & 2 & 3 \\ 1 & 5 & 2 \\ 2 & 1 & 4 \end{pmatrix}$$

Tekintsük a $\bar{\varphi}$ ($\bar{\varphi}(1) = 2,\ \bar{\varphi}(2) = 3,\ \bar{\varphi}(3) = 1$) permutációt. Mivel ez lehetséges megoldás, ezért tekinthetjük úgy, mint az eddigi legjobb lehetséges megoldást. Számítsuk ki a hozzátartozó kvadratikus célfüggvényértéket:

$$a_{11}b_{22} + a_{12}b_{23} + a_{13}b_{21} = 3 \cdot 5 + 2 \cdot 2 + 2 \cdot 1 = 21,$$

$$a_{21}b_{32} + a_{22}b_{33} + a_{23}b_{31} = 2 \cdot 1 + 4 \cdot 4 + 1 \cdot 2 = 20,$$

$$a_{31}b_{12} + a_{32}b_{13} + a_{33}b_{11} = 2 \cdot 2 + 1 \cdot 3 + 3 \cdot 3 = 16.$$

$z(\bar{\varphi}) = 57$. Legyen $x^* = \bar{\varphi}$ és $\bar{z} = 57$.

Most számítsuk ki a **C** és **F** mátrixokat.

$$(a_{ii}b_{jj}) = \begin{pmatrix} 9 & 15 & 12 \\ 12 & 20 & 16 \\ 9 & 15 & 12 \end{pmatrix}$$

Továbbá $\alpha_1 = 2$, $\alpha_2 = \alpha_3 = 1$ és $\beta_1 = \beta_2 = 1$, $\beta_3 = 2$. Most kivonva az α_j, β_j értékeket a megfelelő oszlopok elemeiből, az alábbi $\bar{\mathbf{A}}$ és $\bar{\mathbf{B}}$ mátrixokat kapjuk:

$$\bar{\mathbf{A}} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad \bar{\mathbf{B}} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Ezek után képezzük az alábbi mátrixokat:

$$(\bar{A}_i\beta_j) = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 2 \end{pmatrix} \qquad (\alpha_i\bar{B}_j) = \begin{pmatrix} 2 & 2 & 2 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$(n-1)(\alpha_i\beta_j) = 2 \cdot \begin{pmatrix} 2 & 2 & 4 \\ 1 & 1 & 2 \\ 1 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 4 & 8 \\ 2 & 2 & 4 \\ 2 & 2 & 4 \end{pmatrix}$$

Ekkor

$$(\bar{A}_i\beta_j + \alpha_i\bar{B}_j + (n-1)\alpha_i\beta_j) = \begin{pmatrix} 6 & 6 & 10 \\ 4 & 4 & 7 \\ 4 & 4 & 7 \end{pmatrix}.$$

A kapott mátrixhoz hozzádva az $(a_{ii}b_{jj})$ mátrixot megkapjuk a keresett $\mathbf{C} = (c_{ij})$ mátrixot, amelyre

$$(c_{ij}) = \begin{pmatrix} 15 & 21 & 22 \\ 16 & 24 & 23 \\ 13 & 19 & 19 \end{pmatrix}$$

Most minden $i, j \in \{1, 2, 3\}$ indexpárra az $\bar{\mathbf{A}}$ mátrix i-edik sorvektorának, valamint a $\bar{\mathbf{B}}$ mátrix j-edik sorvektorának elemeivel képezett $(\bar{a}_{ik}\bar{b}_{jl})$ költségmátrixú speciális hozzárendelési feladatot a 14.3.1. segédtétel alapján megoldva, jelöljük a kapott optimimértéket f_{ij}-vel. Megoldva a $3^2 = 9$ darab speciális hozzárendelési feladatot, az alábbi $\mathbf{F} = (f_{ij})$ mátrixot kapjuk:

$$(f_{ij}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Következésképp, az általános hozzárendelési feladat költségmátrixa a következő:

$$\mathbf{F} + \mathbf{C} = (f_{ij}) + (c_{ij}) = \begin{pmatrix} 16 & 21 & 22 \\ 16 & 24 & 23 \\ 13 & 19 & 19 \end{pmatrix}$$

Most oldjuk meg a kapott hozzárendelési feladatot magyar módszerrel. Az eljárás során előálló mátrixsorozat a következő:

$$\begin{pmatrix} 0 & 5 & 6 \\ 0 & 8 & 7 \\ 0 & 6 & 6 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad \overset{+}{\begin{pmatrix} 0^* & 0 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 0^* \end{pmatrix}} \quad \overset{\oplus}{\begin{pmatrix} 0^* & 0' & 0 \\ 0' & 3 & 1 \\ 0 & 1 & 0^* \end{pmatrix}} \overset{+}{+}$$

A $(2,1), (1,1), (1,2)$ indexű elemeket tartalmazó lánc alapján az alábbi 3-elemű független 0-rendszert kapjuk:

$$\begin{pmatrix} 0 & 0^* & 0 \\ 0^* & 2 & 1 \\ 0 & 0 & 0^* \end{pmatrix}$$

azaz a tekintett hozzárendelési feladat optimális megoldása azon φ_0 permutáció, amelyre $\varphi_0(1) = 2$, $\varphi_0(2) = 1$ és $\varphi_0(3) = 3$. A hozzárendelési feladat optimumértékére pedig $z(\varphi_0) = 56$ adódik. Így 56 lesz a B&B fa gyökerében a korlát.

Mivel φ_0 lehetséges megoldás, ezért aktualizálnunk kell a x^* és $\bar{z}$ változók értékeit. A φ_0-hoz tartozó kvadratikus célfüggvényérték a következő:

$$a_{11}b_{22} + a_{12}b_{21} + a_{13}b_{23} = 3 \cdot 5 + 2 \cdot 1 + 2 \cdot 2 = 21,$$

$$a_{21}b_{12} + a_{22}b_{11} + a_{23}b_{13} = 2 \cdot 2 + 4 \cdot 3 + 1 \cdot 3 = 19,$$

$$a_{31}b_{32} + a_{32}b_{31} + a_{33}b_{33} = 2 \cdot 1 + 1 \cdot 2 + 3 \cdot 4 = 16.$$

Így a kvadratikus célfüggvény értéke φ_0-ra $21 + 19 + 16 = 56$, azaz $\bar{z} = 56$.

Most vegyük észre, hogy az eddigi legjobb lehetséges megoldáshoz, φ_0-hoz tartozó kvadratikus célfüggvényértéknél nem kisebb az illető szögponthoz (ez most a gyökér) tartozó alsó korlát. Ez azt jelenti, hogy a tekintett szögponthoz tartozó lehetséges megoldások között nincs jobb, mint φ_0, de mivel a tekintett szögpont a B&B fa gyökere, az összes lehetséges megoldás között nincs jobb, mint φ_0, azaz φ_0 egy optimális megoldása a feladatnak. (A B&B eljárásban a fentiek úgy jelennek meg, hogy $F_0 = \emptyset$.)

15. Ütemezési feladatok

Az ütemezési feladatok vizsgálata az 50-es évek elején kezdődött, majd tekintettel a feladat gyakorlati fontosságára sok különböző modell tanulmányozására került sor, és a témakör nagyon gyors és nagy fejlődésen ment át. A teljesség igénye nélkül megadjuk a [8], [37], [80], [121] összefoglaló jellegű munkákat. Nagyon friss összefoglalás található a [25], [33], [173] munkákban. A modellek nagy számára jellemző, hogy 1977-ben A. H. G. Rinnooy Kan [153] egy konferencia szekciójának összefoglalójában 9000-re becsülte a katalogizálható különböző determinisztikus ütemezési problémák számát. Azóta ez a szám még számottevően növekedett. Tekintettel a témakör nagyságára és bonyolultságára a jelen fejezetben csak vázoljuk az ütemezési problémákat és csak néhány egyszerű modellt vizsgálunk részletesebben. A legalapvetőbb változatok osztályozása után néhány példát mutatunk, miként írhatóak fel ütemezési problémák matematikai programozási feladatokként. Ezt követően bemutatunk néhány egyszerű problémát, amelyekhez létezik polinomiális idejű megoldó algoritmus. Ebben a részben ismertetjük a legrövidebb végrehajtási idő elvén alapuló algoritmusokat és egy példát adunk a legkorábbi határidő elvének használhatóságára. Majd az NP-nehéz modellek közül a legegyszerűbb modellre ismertetünk heurisztikus algoritmusokat, megvizsgáljuk a Lista és az LPT algoritmusokat, meghatározzuk az approximációs hányadosukat. Végül egy bonyolultabb osztályt vizsgálunk a shop ütemezési problémák osztályát.

Mielőtt ismertetnénk az ütemezési problémák egy lehetséges osztályozását, az ütemezés problematikáját egy egyszerű gyakorlati problémán keresztül demonstráljuk.

Három fiú, András (A), Béla (B) és Csaba (C) egy lakást bérelnek és közösen négy újságot, Magyar Nemzet (M), Népszava (N), Pesti Hírlap (P), Új Tükör (U) járatnak. Szombatonként reggel 9 órától mindannyian újságot olvasnak. András M, U, P, N sorrendben olvassa az újságokat, rendre 20, 10, 5, 15 percig, Béla P, N, U sorrendben csak három újságot olvas rendre 30, 10, 15 percig, Csaba pedig M, U, N, P sorrendben olvassa az újságokat rendre 10, 20, 15, 30 percig. A fenti adatokat az alábbi táblázatban összesítettük. A fiúk az újságok olvasása során betartják a következőket:

(1) Valamennyien, ha elkezdenek egy újságot olvasni, azt nem szakítják meg.

(2) Egynél több személy egyidejűleg nem olvassa ugyanazt az újságot.

	sorrend	percek
A olvasása	M, U, P, N	20, 10, 5, 15
B olvasása	P, N, U	30, 10, 15
C olvasása	M, U, N, P	10, 20, 15, 30

15.1. táblázat. Újságolvasási adatok.

A vázolt szituációval kapcsolatosan jogosan vethető fel a következő kérdés.

Az újságok olvasásának milyen időrendi beosztása (ütemezése) eredményezi a közös olvasás teljes idejének minimumát?

Ennek kapcsán az első kérdés az, hogy miként lehet egy lehetséges megoldást leírni? Egy lehetséges technika, hogy minden újságra megadjuk azt, hogy az illető újság milyen sorrendben kerül a fiúkhoz olvasásra. Jó esetben ez nem mond ellent a részolvasások sorrendjének. Ehhez a sorrendhez meg lehet határozni egy időbeli ütemezést, és kapunk egy lehetséges megoldást. Ez a megoldás jól szemléltethető az úgynevezett Gannt diagrammal, amelyben vízszintesen minden újságra felveszünk egy időtengelyt és erre felvisszük az egyes újságok olvasásának időintervallumait. Például tegyük fel, hogy

M-et A, C sorrendben,

N-et B, A, C sorrendben,

P-t B, A, C sorrendben,

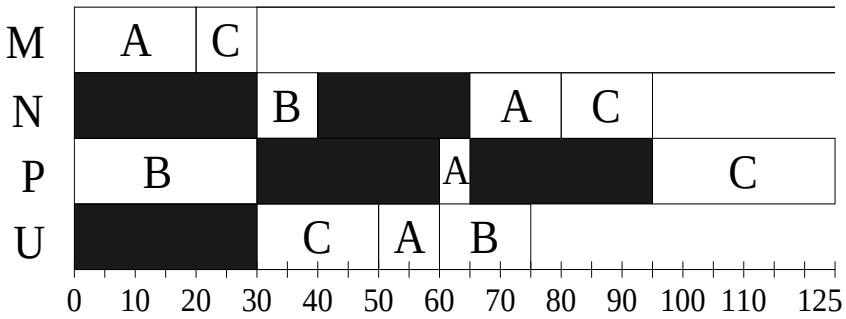
U-t C, A, B sorrendben

olvassák az érintettek. Akkor egy lehetséges ütemezés Gannt diagrammját a 15.1. ábra adja meg.

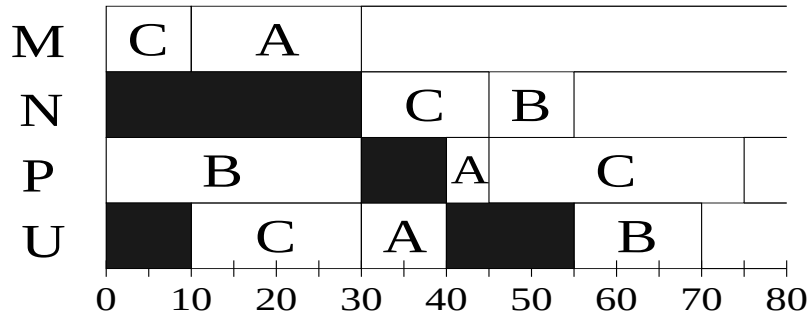
Az alábbi sorrend egy másik ütemezéshez vezet, amit a 15.2. ábrán adtunk meg.

M-re C, A sorrend, N-re C, B, A sorrend, P-re B, A, C sorrend és U-ra C, A, B sorrend.

Vegyük észre, hogy a második ütemezés egyben optimális megoldás is mivel C rövidebb idő alatt nem tudja elolvasni az újságokat.



15.1. ábra. Az olvasás első ütemezése.



15.2. ábra. Az olvasás második ütemezése.

Azt, hogy bizonyos sorrendek ellentmondáshoz vezethetnek, a fiúk példáján demonstráljuk. Ehhez tekintsük az alábbi sorrendeket:

M-re C, A; N-re B, A, C; P-re C, A, B; U-ra B, A, C.

Ekkor B olvasási sorrendjének megfelelően, P olvasásával kezdi a napot, majd az N, U újságokat olvassa. Az egyes újságokra megadott sorrendek alapján N-t és U-t B olvassa elsőként, így ezek az újságok a többiek számára nem elérhetőek, amíg B nem olvasta el őket. Másrészt P-t C olvassa elsőként még B előtt. Mielőtt erre sor kerülne, C-nek ki kell olvasnia N-t és U-t, de ezek elérhetetlenek, így ilyen sorrendi feltételek mellett nem realizálható az újságok elolvasása.

Ütemezési modellek

Az ütemezési problémákban adottak bizonyos páronként különböző gépek és m számú munka, amelyeket az $1, \ldots, m$ számokkal fogunk sorszámozni. A feladat az, hogy ütemezzük az egyes munkák végrehajtását a gépeken, úgy, hogy valamely cél szerint optimális ütemezést kapjunk. A *munkák*

végrehajtásának ütemezésén vagy egyszerűbben a *munkák ütemezésén* azt értjük, hogy a j-edik munkát hozzárendeljük valamely géphez egy S_j *kezdési* és C_j *befejezési idővel* minden j-re. A munkákhoz minden modellben tartozik egy *végrehajtási idő*, amit p_j-vel szokás jelölni. Ez azt adja meg, hogy mennyi ideig tart a munkát elvégezni. Ennek megfelelően a munkához rendelt kezdési és befejezési időkre a $C_j - S_j = p_j$ feltételnek kell teljesülni.

A tekintett példában a munkák a fiúknak az egyes újságokra vonatkozó olvasási tevékenységei. Gépeknek az újságokat tekinthetjük abban az értelemben, hogy a munkákat ezeken kell végrehajtani.

A különböző modelleket többféle szempontból is osztályozhatjuk. A továbbiakban a legismertebb változatokat igyekszünk összegyűjteni.

A munkák meghatározó paraméterei

Mint már említettük minden munkához tartozik egy végrehajtási idő, de más modellekben egyéb paraméterei is vannak az egyes munkáknak.

- A munkákhoz rendelhető *érkezési idő* is, ezt a paramétert a j-edik munkára általában r_j jelöli. Ez az idő azt az időpontot adja meg, amelytől kezdve a munka végrehajtása elkezdhető, tehát a munkára az $S_j \geq r_j$ feltételnek kell teljesülni.

- A j-edik munkához tartozhat egy d_j *határidő*. Itt két különböző típusú modellt vizsgálhatunk. Az első esetben csak olyan ütemezéseket fogadunk el, amelyekre $C_j \leq d_j$, azaz amelyek betartják a határidőt, a másodikban megszeghetjük a határidőt, de ekkor a célfüggvényben a határidő is szerepel.

- A j-edik munkához hozzárendelhetünk egy w_j *súlyt* vagy egy $f_j(t)$ *súlyfüggvényt*, amely azt adja meg mennyire fontos a munka, illetve azt, hogy mennyire fontos a munka t időpontra történő befejezése.

Egy másik fontos osztályozási szempont az, hogy mi az a függvény, aminek az optimumát keressük. Eszerint a szempont szerint is igen sok lehetséges modell került bevezetésre, a továbbiakban a legfontosabbakat vázoljuk. A célfüggvények két alapvető osztályra bonthatók: a maximum célfüggvényekre és az összeg célfüggvényekre.

- Talán a legismertebb modellek azok, amelyekben az utolsónak befejezett munka befejezési idejét akarjuk minimalizálni, azaz ahol a célfüggvény $\max\{C_j : 1 \leq j \leq m\}$, amelyet a lehetséges ütemezésekre minimalizálunk. Szintén gyakran használt célfüggvény a *befejezési idők összegfüggvénye*. Általánosabb esetben, mikor a munkáknak van súlya vagy súlyfüggvénye, akkor a $\sum_{j=1}^{m} w_j \cdot C_j$ illetve $\sum_{j=1}^{m} f_j(C_j)$ függvényeket minimalizáljuk.

- Amennyiben a munkákhoz határidő is tartozik, akkor a célfüggvény általában a késések minimalizálása. Itt két értéket szokás vizsgálni. Az első a *késési idő* (lateness), amely az $L_j = C_j - d_j$ érték, a másik pedig a *csúszási idő* (tardiness), amely az $T_j = \max\{0, L_j\}$ érték. A két maximum célfüggvény a késési időknek illetve a csúszási időknek a maximuma. Egyéb modellekben ezen értékek összegét illetve súlyozott összegét igyekszünk minimalizálni. Itt érdemes azt a modellt említenünk, ahol a cél az elkésett ($T_j > 0$) munkák számának minimalizálása.

- Amennyiben érkezési idők is vannak szokás a befejezési idő helyett a *folyási időt* (flow time) vizsgálni, amely az $F_j = C_j - r_j$ érték. Ezekben a modellekben a célfüggvény ezen F_j értékek maximuma vagy súlyozott összege.

A fenti alapvető osztályozáson kívüli egyéb változatokat, általánosításokat kaphatunk néhány extra feltétellel. Az alábbiakban ezekből gyűjtöttünk össze néhányat.

- Feltehetjük, hogy bizonyos munkákat csak más munkák után lehet elvégezni. Igen sok gyakorlati problémánál előfordulnak ilyen feltételek. Ekkor az egyes munkákhoz tartozik az a feltétel is, hogy mely munkák előzetes végrehajtását követelik meg. Ezen extra feltételek mellett az összes, a fentiekben említett modell vizsgálható. Megemlítjük, hogy ilyen extra feltétellel kezelhető a tekintett példában, hogy a fiúk egy adott sorrendben olvassák az újságokat.

- Az eddigiek során végig azzal a feltétellel éltünk, hogy minden egyes munkához pontosan egy végrehajtási idő tartozik és ez független attól, hogy melyik gépen kerül a munka végrehajtásra. Ez általában gyakorlati problémáknál nem így van. Egy általánosabb modellben minden munkához egy *végrehajtási vektor* tartozik, amely i-edik komponense megadja, hogy az M_i gépen mennyi ideig tart a munkát végrehajtani.

Itt érdemes két speciálisabb esetre kitérni. Az egyik esetben a j-edik munka végrehajtási vektorának i-edik komponense p_j/v_i, ahol v_i az M_i gép sebességének felel meg. A második esetben a korlátozott hozzárendelési esetben a végrehajtási vektor néhány komponense végtelen a többi megegyezik, ez azt jelenti, hogy a gépek azonosak csak a munka néhány gépen nem hajtható végre.

- Egy másik általánosítás, amelyben megengedjük, hogy a munkák végrehajtása megszakítható legyen. Ekkor a j-edik munkához nem egy darab legalább p_j hosszú intervallumot kell hozzárendelnünk valamely gépen, hanem több, egymást nem átfedő intervallumot (akár különböző gépeken), amelyek összhossza legalább p_j.

A fentieken kívül még rengeteg változat létezik, de terjedelem hiányában nem térhetünk ki minden modellre és a jegyzet célja nem is ezek részletes összegyűjtése. A különböző változatok részletesebb osztályozása, és az egyes modelleket leíró három részből álló jelölésrendszer megtalálható a [173], [33] dolgozatokban. A jelölésrendszer három mezőt használ a modellek megadására, az első mező tartalmazza a gépek számára, típusára vonatkozó információkat, a második mező a modellre vonatkozó egyéb feltételeket, a harmadik mező pedig a célfüggvényt adja meg. Jelen fejezetben a modellek megadására használjuk a jelölésrendszert is, de a modelleket minden esetben definiáljuk részletesen is.

Felírás matematikai programozási feladatként

Több ütemezési probléma felírható matematikai programozási feladatként (lineáris programozási feladat, egészértékű programozási feladat), és így az ezen általános problémák megoldására kidolgozott eljárások használhatóak az adott ütemezési feladat megoldására is. Ebben a részben néhány ilyen példát mutatunk be.

Felírás lineáris programozási feladatként

Egyes ütemezési problémák felírhatók lineáris programozási feladatként. Példaként a $Pn|prmp|C_{max}$ problémát tekintjük, (n párhuzamos azonos gépen ütemezzük a munkákat a maximális befejezési időt minimalizálva, a munkák megszakítását engedélyezve).

Jelölje az $x_{ij} \geq 0$ változó, azt az időmennyiséget, amelyet az i-edik gép összesen a j-edik munkával tölt. Ekkor a feladat a következő formában írható fel:

$$\sum_{i=1}^{n} x_{ij} = p_j, \quad j = 1, \ldots, m,$$
$$\sum_{i=1}^{n} x_{ij} \leq C_{max}, \quad j = 1, \ldots, m,$$
$$\sum_{j=1}^{m} x_{ij} \leq C_{max}, \quad i = 1, \ldots, n,$$
$$x_{ij} \geq 0, \quad i = 1, \ldots, n, j = 1, \ldots, m$$

.

———————————————————————

$$C_{max} \to \min$$

Ekkor az első csoportja a feltételeknek azt adja meg, hogy összesen a gépek a j-edik munkával valóban p_j időt töltenek. A második feltételcsoport azt biztosítja, hogy a teljes idő amelyet egy munkán a gépek összesen dolgoznak legfeljebb a maximális befejezési idő (ezeket a feltételeket helyettesíthetjük a $p_j \leq C_{max}$ feltételekkel). A harmadik csoportja a feltételeknek azt biztosítja, hogy egyetlen gép sem dolgozik többet, mint C_{max}.

Felírás egészértékű programozási feladatként

Sok ütemezési probléma megadható egészértékű programozási feladatként. Példaként az $1||\sum w_j C_j$ problémát tekintjük azt a problémát, amelyben egyetlen gép van, a munkákat két paraméter határozza meg, a végrehajtási idő és a munka súlya, és amelyben a cél minimalizálni a teljes súlyozott befejezési időt ($\sum w_j C_j$). Két különböző reprezentációt is megadunk.

Legyen x_{jk} egy döntési változó, amely 1, ha a j munka az ütemezésben hamarabb kerül végrehajtásra, mint k és 0 egyébként. Ezen változók segítségével az alábbi programozási feladat írja le az ütemezési problémát.

$$x_{kj} + x_{jk} = 1, \quad j, k = 1, \ldots, m, j \neq k,$$

$$x_{kj} + x_{jl} + x_{lk} \leq 2 \quad j, k, l = 1, \ldots, m, j \neq k, k \neq l, l \neq j,$$

$$x_{jk} \in \{0, 1\} \quad j, k = 1, \ldots, m,$$

$$x_{jj} = 0 \quad j = 1, \ldots, m$$

.

———————————————————————

$$\sum_{j=1}^{m} \sum_{k=1}^{m} w_j p_k x_{kj} + \sum_{j=1}^{m} w_j p_j \to \min$$

Mivel a j-edik munka befejezési ideje $\sum_{k=1}^{m} p_k x_{kj} + p_j$, ezért a célfüggvény valóban $\sum w_j C_j$, és az is könnyen ellenőrizhető, hogy a többi feltétel pontosan azt írja le, hogy az x_{jk} változók a munkák egy sorbarendezését adják meg. Érdemes megjegyezni, hogy a fenti programozási feladat könnyen kiterjeszthető arra az esetre, amelyben a munkákra precedencia feltételek vannak előírva, minden ilyen feltétel rögzíti valamely x_{jk} változó értékét.

A fenti típusú interpretáció, ahol döntési változók a munkák sorrendjét adják meg, nem terjeszthető ki egynél több gép esetére. Az alábbiakban egy olyan egészértékű programozási reprezentációt adunk meg, amely kiterjeszthető több gép esetére. Ezen reprezentáció használatához fel kell tennünk, hogy a végrehajtási idők egészek. Könnyen adódik, hogy ekkor elegendő azokat a lehetséges ütemezéseket vizsgálni, amelyekben a kezdési idők is egészek. Továbbá fontos megjegyeznünk, hogy az alábbi egészértékű programozási feladatban a változók száma rendkívül nagy lehet.

Legyen az x_{jt} változó 1, ha a j-edik munka a t egész időpontban kezdődik, 0 különben. Legyen $l = \sum_{j=1}^{m} p_j - 1$. Ekkor az alábbi programozási feladat írja le az ütemezési problémát:

$$\sum_{t=0}^{l} x_{jt} = 1, \quad j = 1, \ldots, m,$$

$$\sum_{j=1}^{m} \sum_{s=\max\{t-p_j,0\}}^{t-1} x_{js} = 1, \quad t = 0, 1, \ldots, l,$$

$$x_{jt} \in \{0, 1\}, \quad j = 1, \ldots, m, t = 0, \ldots, l.$$

$$\sum_{j=1}^{m} \sum_{t=1}^{l} w_j (t + p_j) x_{jt} \to \min$$

A változók fentiekben megadott értelmezése mellett egyből adódik, hogy a célfüggvény valóban a teljes súlyozott befejezési idő. Az első feltételcsoport azt biztosítja, hogy minden munkára pontosan egy kezdési időt határozunk meg. A második feltételcsoport pedig azt biztosítja, hogy minden időpontban csak egy munkát hajtunk végre a gépen.

Egyszerű megoldó elvek

Az első fontos észrevételünk az ütemezési problémákkal kapcsolatban az, hogy reguláris célfüggvények esetén (*regulárisnak* nevezzük azokat a célfüggvényeket, amelyek monoton növekvőek a befejezési időkben) elegendő azon ütemezéseket vizsgálnunk, amelyekben nincs üres idő, azaz amelyekben a gépek megszakítás nélkül dolgoznak. Ezt a nyilvánvaló állítást sokszor fogjuk használni az ütemezési problémák vizsgálata során.

Legrövidebb végrehajtási idő elve

Első példaként az $1||\sum w_jC_j$ problémát tekintjük (egyetlen gépen ütemezzük a munkákat és célunk a befejezési idők súlyozott összegét minimalizálni). Ezen célfüggvény esetén a w_j/p_j hányados tekinthető a j-edik munka egységenkénti fontosságának, így természetes gondolat azon munkákat ütemezni először, amelyekre ez a hányados nagyobb. Ezt az elvet "súlyozott legrövidebb végrehajtási idő" elvének nevezzük (a WSPT rövidítést használjuk az angol "weighted shortest processing time" kifejezés alapján). Az algoritmust, amely ezen az elven alapul a [166] dolgozatban mutatták be. Az eljárás mindig azt a munkát ütemezi következőnek, amelynek az egységnyi fontossága maximális (ha több ilyen is van a legkisebb indexűt). Az algoritmust a következő példán ismertetjük:

15.1. példa. Vegyük azt a problémát, amelyben a következő három munka van $J_1 = (2,4), J_2 = (3,2), J_3 = (2,3)$, ahol a munkákat (w_j, p_j) értékpárral adtuk meg. Ekkor a munkáknak a w/p értékek szerinti sorbarendezése J_2, J_3, J_1, így az algoritmus a munkákat ebben a sorrendben hajtja végre üres idő nélkül, J_2-t a $(0,2)$, J_3-at a $(2,5)$ és végül J_1-et az $(5,9)$ időintervallumban.

Miként a következő állítás mutatja ez a megközelítés valóban optimális a vizsgált probléma esetén.

15.1. tétel. *A WSPT elv optimális ütemezést ad az* $1||\sum w_jC_j$ *problémára.*

Bizonyítás. Elsőként igazoljuk, hogy csak a WSPT elvet követő algoritmusok adhatnak optimális ütemezést. Ehhez tegyük fel, hogy van egy olyan optimális S ütemezés, amelyre nem teljesül a WSPT elv. Mivel nem teljesül az elv, ezért van az S ütemezésben két egymást követő munka, J_i és közvetlenül utána J_k, amelyekre

$$\frac{w_i}{p_i} < \frac{w_k}{p_k}.$$

Jelöljük t-vel a J_i munka kezdési idejét az S ütemezésben. Tekintsük azt az S' ütemezést, amelyben a J_i és a J_k munkákat felcseréljük. Az S ütemezésben J_i befejezési ideje $t+p_i$, J_k befejezési ideje pedig $t+p_i+p_k$. Az S' ütemezésben J_k befejezési ideje $t+p_k$ és J_i befejezési ideje $t+p_i+p_k$. A többi munka befejezési ideje ugyanaz mindkét ütemezésben. Jelölje a többi munkára számított $\sum w_j C_j$ összeget Q. Ekkor az S ütemezés teljes költsége $Q + w_i(t+p_i) + w_k(t+p_i+p_k)$, az S' ütemezés teljes költsége $Q + w_k(t+p_k) + w_i(t+p_i+p_k)$. Másrészt

$$\frac{w_i}{p_i} < \frac{w_k}{p_k}, \quad \text{azaz} \quad w_i p_k < w_k p_i,$$

amiből azt kapjuk, hogy az S' ütemezés költsége kisebb az S ütemezés költségénél. Ez ellentmond S optimalitásának, amivel az állítását igazoltuk.

Másrészt a fentiekhez teljesen hasonlóan látható, hogy amennyiben két egymást követő munkára egyenlő a w/p hányados, akkor a felcserélésükkel kapott ütemezésre a célfüggvény értéke nem változik. Mivel a feladatnak véges sok lehetséges megoldása van, így létezik optimális megoldás, ezért a fentiek alapján következik a tétel állítása.

Amennyiben a problémában az egyes munkáknak nincsenek súlyai, akkor a WSPT elv az SPT elvre redukálódik, amely szerint mindig a legkisebb végrehajtási idővel rendelkező munkát ütemezzük először. A fenti tétel alapján következik, hogy az SPT elv optimális az $1||\sum C_j$ probléma esetén. Az alábbiakban adunk egy újabb bizonyítást erre az állításra, amely bizonyítás előnye, hogy kiterjeszthető, a párhuzamos gépek esetére is.

15.2. tétel. *Az SPT elv optimális ütemezést ad az* $1||\sum C_j$ *problémára.*

Bizonyítás. Tekintsünk egy gépet, jelölje a gépen ütemezett munkákat növekvő kezdési idő szerint $J_1, J_2, \ldots, J_n$. Ekkor ezen gépre a $\sum C_j$ összeg $np_1 + (n-1)p_2 + \ldots + p_n$. Tehát feladatunk az, hogy az $1, \ldots, n$ együtthatókat hozzárendeljük a $p_1, \ldots, p_n$ értékekhez oly módon, hogy a fenti összeg minimális legyen. A kvadratikus hozzárendelési feladat tárgyalása során igazolt 14.3.1. segédtétel alapján adódik, hogy az együtthatók és a p_i értékek azon párosítására lesz minimális az összeg, amelyben az együtthatók és a p_i értékek ellentétesen vannak rendezve (a legnagyobb együtthatóhoz rendeljük a legkisebb értéket, a második legnagyobbhoz a második legkisebb értéket és így folytatjuk). Másrészt ez a hozzárendelés pontosan az SPT ütemezési elvet írja le.

Most tekintsük a párhuzamos gépek esetét. Ebben az esetben az SPT elv következő kiterjesztése lesz a megfelelő ütemezési stratégia. Tegyük fel,

hogy m/n egész. Ez elérhető, ha a munkák sorozatát kiegészítjük fiktív 0 végrehajtási idejű munkákkal. Az MSPT (módosított SPT) elv sorbarendezi a munkákat növekvő végrehajtási idő szerint, ezt követően az első n munka lesz az n darab gépen elsőként végrehajtva, a második n munka másodikként és így folytatva az utolsó n munka m/n-edikként. Az eljárást az alábbi példán szemléltetjük.

15.2. példa. Legyen a gépek száma 3 és tegyük fel, hogy a következő 5 munkánk van: $J_1 = 4, J_2 = 1, J_3 = 1, J_4 = 2, J_5 = 3$, ahol a munkákat a végrehajtási idejükkel adtuk meg. Ekkor a sorozatot kiegészítjük egy fiktív 0 végrehajtási idővel rendelkező J_6 munkával. Sorbarendezzük őket a kapott sorrend $J_6, J_2, J_3, J_4, J_5, J_1$. Az algoritmus az első gépen a J_6 fiktív munkát és a J_4 munkát ütemezi, ez utóbbit a $(0, 2)$ időintervallumban, a második gépen a J_2 és J_5 munkákat a $(0, 1)$ és $(1, 4)$ időintervallumokban, a harmadik gépen a J_3 és J_1 munkákat a $(0, 1)$ és $(1, 5)$ időintervallumokban.

Az algoritmus valóban optimális megoldást ad, miként a következő állítás mutatja.

15.3. tétel. *Az MSPT elv optimális ütemezést ad a* $Pn||\sum C_j$ *probléma esetén.*

Bizonyítás. Vegyünk n gépet. Legyenek az i-edik gépen $(i = 1, \ldots, n)$ ütemezett munkák $J_{i1}, J_{i2}, \ldots, J_{ik_i}$. Ekkor $\sum_{i=1}^{n} k_i = m$ és az ütemezésre a $\sum C_j$ összeg $\sum_{i=1}^{n} \sum_{j=1}^{k_i} (k_i + 1 - j) p_{ij}$. A fenti összegben, adott számokhoz (a p_{ij} végrehajtási idők) rendelünk hozzá együtthatókat. Összesen m pozitív egész együtthatónk van, amelyekből minden l pozitív egészre legfeljebb n együttható veheti fel az l értéket. Mivel a végrehajtási idők nemnegatívak, ezért könnyen látható, hogy a teljes befejezési időt megadó összeg minimális, ha az együtthatók között minden $i = 1, 2, \ldots, n/m$ értékre n darab veszi fel az i értéket, és a végrehajtási idők és az együtthatók ellentétesen vannak rendezve. Másrészt ez pontosan az MSPT ütemezési elv által kapott ütemezésnek felel meg.

Legkorábbi határidő elve

Amennyiben a munkák paraméterei között határidők is szerepelnek, és a cél valamely határidőtől függő függvény minimalizálása, akkor gyakran használt elv, amely azon munkákat ütemezi elsőként amelyeknek a legkorábbi a határideje, ezt az elvet EDD (earliest due date) jelöli. Mielőtt példát mutatnánk olyan problémára, amelynél az EDD elv optimális ütemezést ad, egy

általános algoritmust mutatunk be, amely speciális esetben az EDD elvre redukálódik. Ezt az általános algoritmust a [120] dolgozatban ismertették.

Az algoritmus több célfüggvény esetén is használható. Legyen a j-edik munkára h_j egy monoton növekvő függvény, továbbá legyen $h_{\max} = \max\{h_1(C_1), h_2(C_2), \ldots, h_m(C_m)\}$. Ezen célfüggvény speciális esetként tartalmazza a $\max L_j$, és $\max T_j$ függvényeket, mivel mind L_j mind T_j monoton növekvő függvényei a befejezési időnek. A vizsgált modellben megengedjük precedencia feltételek megadását is. Tehát a probléma, amit vizsgálunk az $1|\text{prec}|h_{max}$ jelöléssel írható le.

Az alábbiakban bemutatott algoritmus hátulról, az utolsónak végrehajtott munka felől építi fel az ütemezést. J mindig a már elhelyezett munkák halmaza, J^C azon munkák halmaza, amelyeket még el kell helyeznünk (a J halmaz komplementere) és J' azon J^C-beli munkák halmaza, amelyek az adott lépésben ütemezhetők, azaz azon munkáké, amelyekre az összes precedencia feltételek szerinti rákövetkező munkát már elhelyeztük az ütemezés végén.

Lawler eljárása [120]

- *1. lépés:* Legyen $J = \emptyset$, $J^C = \{1, \ldots, m\}$, és J' azon munkák halmaza, amelyekre nincs rákövetkező munka a precedencia gráfban.
- *2. lépés:* Legyen $j^* \in J'$ olyan munka, amelyre

$$h_{j^*}(\sum_{j \in J^C} p_j) = \min_{i \in J'}(h_i(\sum_{j \in J^C} p_j)).$$

  Egészítsük ki J-t j^*-gal, töröljük j^*-ot a J^C halmazból. Legyen J' az ütemezhető munkák új halmaza .
- *3. lépés:* Ha $J^C = \emptyset$, az algoritmus véget ért, egyébként folytassuk a 2. lépéssel.

15.4. tétel. *Az algoritmus a munkák egy optimális ütemezését adja az* $1|\text{prec}|h_{\max}$ *problémára.*

Bizonyítás. Az állítást indirekt igazoljuk. Ehhez tegyük fel, hogy a kapott ütemezés nem optimális. Jelölje S az algoritmus által adott ütemezést. Tekintsük azt az optimális ütemezést, amelyhez olyan sorrendje tartozik a munkáknak, amely sorrendnek a legtöbb utolsó közös munkája van S-sel. Jelölje ezt OPT. Mivel a tekintett ütemezés nem S, ezért ez az ütemezés,

valamely iterációban S-től különböző munkát választ. Tegyük fel, hogy ebben a lépésben OPT a k munkát választja (ekkor a $k \in J'$ feltételnek kell teljesülnie az adott lépésben). Másrészt S egy olyan j^* munkát választ, amely minimalizálja a $h_i(\sum_{j \in J^C} p_j)$ értékeket az $i \in J'$ egészekre. Ez azt jelenti, hogy OPT-ban j^* a k munka előtt kerül ütemezésre.

Vegyük azt az ütemezést, amelyet úgy kapunk OPT-ból, hogy j^*-ot a jelenlegi helye helyett közvetlenül k után ütemezzük, a többi munka sorrendjét változatlanul hagyva. Azokat munkákat, amelyek az eredeti ütemezésben j^* és k között voltak továbbá k-t hamarabb fejezzük be. Egyedül a j^* munka befejezési ideje nő, de ezen munka minimalizálta a $h_i(\sum_{j \in J^C} p_j)$ értéket az $i \in J'$ halmazon, így az általa okozott költség nem nagyobb, mint k költsége az eredeti ütemezésben. Következésképp a j^* elemet beillesztve a k elem mögé, egy olyan ütemezést kapunk, amelynek a költsége nem nagyobb egy optimális ütemezésnél, így szintén optimális. Másrészt az új ütemezés végén eggyel több munka egyezik meg az S ütemezés utolsó munkáival, amivel ellentmondáshoz jutunk.

Az $1||L_{\max}$ probléma a legismertebb speciális esete a $1|\text{prec}|h_{\max}$ problémának. Ebben az esetben $h_j = C_j - d_j$, az algoritmus pedig a legkorábbi határidő, rövidítve EDD (earlyest due date) szabályra redukálódik, amely azt mondja ki, hogy azt a munkát ütemezzük hamarabb, amelynek hamarabb van a határideje.

Heurisztikus algoritmusok az alapproblémára

Ebben a részben az egyik legegyszerűbb változatot vizsgáljuk. A modellben n darab azonos $M_1, \ldots, M_n$ gépünk van , a $J_1, \ldots, J_m$ munkákhoz pedig csak egyetlen paraméter tartozik, a p_j, $j = 1, \ldots, m$ végrehajtási idők. Cél egy olyan ütemezés megkonstruálása, amelyre a befejezési idők maximuma minimális.

A probléma egy gép esetén triviális bármely olyan ütemezésre, amelyben a gép folyamatosan dolgozik, (nem várunk a munkák elvégzése közben) a $\max\{C_j : 1 \leq j \leq m\}$ érték megegyezik a végrehajtási idők összegével. Továbbá az is nyilvánvaló, hogy amennyiben minden munkát elvégzünk, akkor nem fejezhetjük be a munkákat ezen időpont előtt. Ha a gépek száma több, mint 1, akkor a feladat lényegesen nehezebb. Igazolást nyert, hogy $n \geq 2$ esetén a probléma NP-nehéz. Másrészt ebben az esetben egy lehetséges ütemezést meghatározhatunk azáltal, hogy az egyes munkákat

mely gépekhez rendeljük hozzá. Amennyiben minden gépre megkapjuk, hogy mi a géphez rendelt munkáknak a halmaza, akkor minden egyes gépre, az ott levő munkákat optimálisan úgy ütemezhetjük, hogy a gép folyamatosan dolgozzon, és minden ilyen ütemezésre ugyanannyi lesz a gépen a maximális befejezési idő, a munkáknak a végrehajtási idejeinek az összege. Ezt az értéket a gép *töltésének* nevezzük. A fogalmat használjuk általánosabb értelemben is, gépek egy halmazának a töltésén a gépekhez rendelt munkák végrehajtási idejeinek az összegének és a gépek számának a hányadosát értjük.

Mivel a feladat NP-nehéz ezért erre a feladatosztályra is fontos heurisztikus algoritmusok kidolgozása. Az alábbiakban két egyszerű heurisztikus algoritmussal ismerkedünk meg.

Az első R. L. Graham-től [79] származó algoritmus, amelyet *lista algoritmusnak* nevezünk a következőképpen működik.

Lista algoritmus ([79])

Előkészítő rész. A J_1 munkát rendeljük az M_1 géphez, továbbá legyen $r := 1$.

Iterációs rész (r-edik iteráció). Ha $r = m$, akkor vége az eljárásnak. Ellenkező esetben a J_{r+1} munkát rendeljük ahhoz a géphez, amely gépen minimális a töltés. Ha több ilyen gép is van válasszuk a legkisebb indexűt.

Az algoritmus az optimálishoz közeli eredményt eredményez, amint azt az alábbi tétel mutatja.

15.5. tétel. *A lista algoritmus approximációs hányadosa* $2 - 1/n$*, ahol* n *a gépek száma.*

Bizonyítás. Elsőként igazoljuk, hogy az algoritmus $2-1/n$-approximációs. Legyen $\sigma = \{J_1, \ldots, J_m\}$ tetszőleges munkasorozat rendre $p_1, \ldots, p_m$ végrehajtási időkkel. Tekintsük a lista algoritmus által kapott ütemezést. Legyen J_l az a munka, amely legkésőbb fejeződik be. Vizsgáljuk ezen munka S_l kezdési idejét. Mivel egyetlen gép sem kezdte el a munkát ütemezni S_l előtt, ezért minden gép szünet nélkül dolgozott az S_l időpontig. Ebből azt kapjuk, hogy

$$S_l \leq \frac{1}{n} \sum_{\substack{j=1 \\ j \neq l}}^{m} p_j = \frac{1}{n}(\sum_{j=1}^{m} p_j - p_l) = \frac{1}{n}(\sum_{j=1}^{m} p_j) - \frac{1}{n} p_l.$$

Következésképp

$$lista(\sigma) = S_l + p_l \leq \frac{1}{n}(\sum_{j=1}^{m} p_j) + \frac{n-1}{n} p_l.$$

Másrészt az optimális ütemezésben is végre kell hajtani az összes munkát, így $OPT(\sigma) \geq \frac{1}{n}(\sum_{j=1}^{m} p_j)$. Továbbá a p_l munkát is végre kell hajtani valamely gépen, így $OPT(\sigma) \geq p_l$. Ezen becslések alapján egyből adódik, hogy

$$lista(\sigma) \leq (1 + \frac{n-1}{n}) OPT(\sigma),$$

amivel bizonyítottuk, hogy az algoritmus $2 - 1/n$-approximációs.

Most igazoljuk, hogy a tekintett korlát éles. Vegyünk $n(n-1)$ darab munkát $1/n$ végrehajtási idővel, majd egy munkát 1 végrehajtási idővel. Ekkor a lista algoritmus az első $n(n-1)$ munkát egyenletesen elosztja a gépek között, majd az utolsó munkát az M_1 gépen ütemezi. Tehát a maximális befejezési idő $1+(n-1)/n$ lesz. Egy optimális ütemezés pedig a rövid munkákat egyenletesen osztja szét az első $n-1$ gép között, majd az utolsó munkát az n-edik géphez rendeli, és a maximális befejezési ideje 1 lesz. Tehát ebben az esetben az algoritmus által kapott megoldás és az optimális megoldás célfüggvényértékeinek hányadosa $2 - 1/n$, amivel igazoltuk az állításunkat.

Amint azt láthattuk a korlát élességének igazolásánál, a lista algoritmus legnagyobb hibája akkor jön elő, amikor sok rövid munka után egy hosszú munkát kell végrehajtani. Ezt a problémát jobban kezeli a következő szintén R. L. Graham-től származó eljárás.

LPT algoritmus ([79])

- *1. lépés.* Rendezzük sorba a munkákat csökkenő végrehajtási idő szerint.

- 2. lépés. A munkák kapott listáján hajtsuk végre a lista algoritmust.

Az első rendezési fázis valóban javítja az eljárás hatékonyságát, amint azt a következő állítás mutatja.

15.6. tétel. *Az LPT algoritmus approximációs hányadosa* $4/3 - 1/(3n)$.

Bizonyítás. Elsőként igazoljuk, hogy az algoritmus $4/3 - 1/(3n)$ - approximációs. Az állítást indirekt igazoljuk. Ehhez tegyük fel, hogy léteznek olyan ellenpéldák, amelyekre az optimális ütemezés és az LPT algoritmus által kapott ütemezés költségeinek hányadosa nagyobb mint $4/3 - 1/(3n)$. Ezen ellenpéldák közül van olyan, amely minimális számú munkát tartalmaz. Tekintsünk egy ilyen ellenpéldát, jelölje m a munkák számát, és σ a munkák listáját. Feltehetjük, hogy $p_1 \geq p_2 \geq \ldots \geq p_m$.

Mivel a tekintett ellenpéldában a munkák száma minimális, ezért az utolsónak ütemezett munka befejezési ideje megegyezik a maximális befejezési idővel. Amennyiben ez a tulajdonság nem teljesülne, akkor arra a σ' inputra, amelyet a tekintett ellenpéldából az utolsó munkát elhagyva kapnánk $LPT(\sigma') = LPT(\sigma)$ és $OPT(\sigma) \geq OPT(\sigma')$ teljesülne. Következésképp egy olyan ellenpéldát kapnánk, amely kevesebb munkát tartalmaz, mint σ, ami ellentmond σ minimalitásának.

A fentiek alapján azt kapjuk, hogy az utolsó munka kezdési ideje $LPT(\sigma) - p_m$. Másrészt eddig az időpontig az összes gép dolgozott, így

$$LPT(\sigma) - p_m \leq \frac{\sum_{i=1}^{m-1} p_i}{n}.$$

Továbbá az optimális ütemezésben is végre kell hajtani az összes munkát, így $OPT(\sigma) \geq \frac{1}{n}(\sum_{j=1}^{m} p_j)$.

Következésképp a következő egyenlőtlenségek teljesülnek a σ ellenpéldára.

$$\frac{4}{3} - \frac{1}{3n} < \frac{LPT(\sigma)}{OPT(\sigma)} \leq \frac{p_m + \sum_{i=1}^{m-1} p_i/n}{OPT(\sigma)} =$$

$$\frac{p_m(1-1/n)}{OPT(\sigma)} + \frac{\sum_{i=1}^{m} p_i/n}{OPT(\sigma)} \leq \frac{p_m(1-1/n)}{OPT(\sigma)} + 1.$$

A fenti egyenlőtlenség jobb és baloldalát véve, a következő korlát adódik az $OPT(\sigma)$ értékre:

$$OPT(\sigma) < 3p_m.$$

Mivel p_m a minimális végrehajtási idő, ezért a fenti egyenlőtlenség azt jelenti, hogy az optimális ütemezésben minden gép legfeljebb két munkát tartalmaz, azaz $m \leq 2n$. Másrészt ebben az esetben az ütemezési probléma a következő feladatra redukálódik. Rendezzünk $2n$ számot párokba úgy, hogy a párokba tartozó számok összegeinek maximuma minimális legyen. Ezen feladat megoldása az, ha az i-edik legnagyobb számot az i-edik legkisebb számmal párosítjuk, amely pontosan az LPT ütemezési eljárásnak felel meg.

Következésképp azt kaptuk, hogy az LPT eljárás a σ ellenpéldán az optimális ütemezést adja, azaz ellentmondáshoz jutottunk, amivel igazoltuk, hogy az algoritmus $4/3 - 1/(3n)$-approximációs.

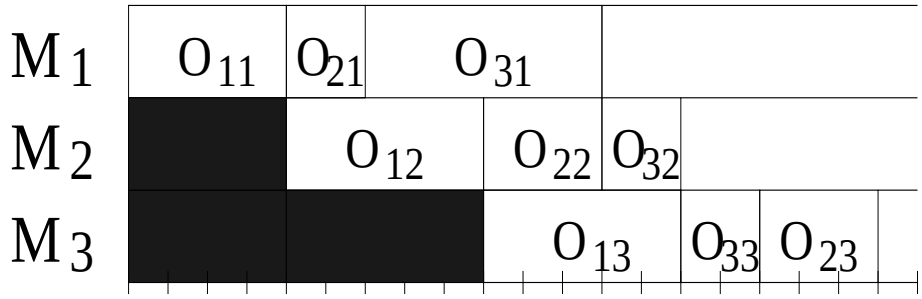
Most igazoljuk, hogy a korlát éles. Legyen n a gépek száma és vegyük azt a σ_n munkasorozatot, amely $2n+1$ munkából áll, amelyekből 3 darab munkának a végrehajtási ideje n, továbbá az $i = n+1, \ldots, 2n-1$ értékekre rendre 2 darab munka van, amelynek a végrehajtási ideje i. Ezen sorozatra az optimális megoldás a 3 darab n hosszú munkát egy gépen ütemezi, a többieket párosítja $n-1$ párba úgy, hogy minden párban $3n$ legyen az összeg. Így $OPT(\sigma_n) = 3n$. Másrészt LPT az első $2n$ munkát párosítja a gépeken úgy, hogy mindenhol $3n-1$ lesz a töltés, majd az utolsó n hosszú munkát az első gépen ütemezi. Tehát $LPT(\sigma_n) = 4n-1$, amivel a korlát élességét igazoltuk.

Itt érdemes megjegyezni a lista algoritmus egy nagy előnyét az LPT algoritmushoz képest, nevezetesen azt, hogy a lista algoritmus abban az esetben is alkalmazható, ha nem ismert az ütemezés elkezdésekor az összes munka, hanem a munkákat egyenként kapjuk meg és az egyes munkákat azonnal a többi munkára vonatkozó ismeretek nélkül kell ütemeznünk. Az ilyen problémákat on-line problémának nevezzük és ezek vizsgálata egy új gyorsan fejlődő terület. Speciálisan az on-line ütemezési problémákról ad áttekintést a [161] dolgozat.

Shop ütemezés

Shop ütemezésről beszélünk abban az esetben, ha a gépeken minden végrehajtandó munka több műveletből áll. Minden egyes művelet egy, a művelethez rendelt gépen hajtható végre, és adott a művelet végrehajtási ideje. Ezen osztályon belül két fő csoportot különböztetünk meg. Amennyiben bármely munkára a hozzátartozó műveletek tetszőleges sorrendben végrehajthatók, akkor *open shop ütemezésről* beszélünk. Ezt a modellt itt nem tárgyaljuk. A másik esetben ismét megkülönböztetünk két modellt. Ha a munkák műveleteihez tartozó gépek sorrendje nem azonos akkor *job shop ütemezésről* beszélünk. (A fiúk újságolvasási problémája így egy job shop ütemezési feladat.) Ha a gépek sorrendje azonos minden munkára, akkor *flow shop ütemezésről* szokásos beszélni. A továbbiakban ezt a problémaosztályt vizsgáljuk. Speciálisabban olyan problémákat tekintünk, amelyekben a gépek száma n és minden J_i munka n számú műveletből, az $O_{i1}, \ldots, O_{in}$ műveletekből áll, amelyek közül a k-adik műveletetet a k-adik gépen kell

végrehajtanunk. A következő Gannt diagram egy flow shop ütemezést mutat.



15.3. ábra. Egy flow shop ütemezés.

Észrevehetjük, hogy a fenti flow shop ütemezési példa esetén a gépeken a munkák végrehajtása nem azonos sorrendben történik. Amennyiben még az azonos sorrendet is kikötjük, akkor *permutációs flow shop* problémáról beszélünk. Amint az elnevezés is mutatja ebben az esetben a munkáknak kell azt a sorrendjét meghatározni, amely sorrendre az ütemezés optimális.

A probléma általában még ennyi kikötés mellett is NP-nehéz marad, de ebben az esetben két gépre már ismert hatékony eljárás. A továbbiakban bemutatjuk ezt az eljárást és igazoljuk annak helyességét.

Jelölje a gépeket M_1 és M_2 a munkákat pedig rendre $J_1, \ldots, J_m$. A J_i $i = 1, \ldots, m$ munka két műveletből áll, először az O_{i1} műveletet kell az M_1 gépen végrehajtani, ami τ_{i1} ideig tart, majd az O_{i2} műveletet az M_2 gépen ami τ_{i2} ideig tart.

Johnson algoritmusa ([100])

- *1. lépés.* Legyen $k := 1$ és $l := m$, és tekintsük a nem ütemezett munkák $J = \{J_1, \ldots, J_m\}$ halmazát.

- *2. lépés.* Keressük meg a $\{\tau_{i1}, \tau_{i2} | J_i \in J\}$ halmaz egy minimális elemét. Jelölje a hozzátartozó indexet i. Ha i nem egyértelműen meghatározott, akkor a lehetséges indexek közül válasszuk a legkisebbet.

- *3. lépés.* Ha a 2. lépésben választott elem τ_{i1} akkor helyezzük a J_i

munkát a listánk k-adik helyére, töröljük a J halmazból, és növeljük k értékét 1-gyel. Majd lépjünk az 5. lépésre.

- *4. lépés.* Ha a 2. lépésben választott elem τ_{i2} akkor helyezzük a J_i munkát a listánk l-edik helyére, töröljük a J halmazból, és csökkentsük l értékét 1-gyel. Majd lépjünk az 5. lépésre.

- *5. lépés.* Amennyiben J üres, akkor vége az eljárásnak. Az optimális ütemezést kapjuk meg, ha az eljárás során meghatározott lista sorrendje szerint hajtjuk végre a munkákat késlekedés nélkül. Ellenkező esetben lépjünk a 2. lépésre.

Az eljárás helyessége egyből következik az alábbi tételből.

15.7. tétel. *Ha egy permutációs flow shop probléma egy késleltetést nem tartalmazó S ütemezésében egy J_i munkára és az ütemezésben utána közvetlenül következő J_l munkára*

$$\min\{\tau_{i1}, \tau_{i2}, \tau_{l1}, \tau_{l2}\} = \min\{\tau_{l1}, \tau_{i2}\}$$

teljesül, akkor arra az S' ütemezésre, amelyet úgy kapunk S -ből, hogy felcseréljük az J_i és J_l munkákat a maximális befejezési idő legfeljebb akkora lesz, mint az S ütemezésben.

Bizonyítás. Elsőként tegyük fel, hogy a segédtételben szereplő értékek közül τ_{l1} minimális. Az az eset, amelyben τ_{i2} minimális teljesen hasonlóan igazolható. Jelölje Q azt az időt, amikor S-ben M_1 elkezdheti a J_i munka ütemezését (a J_i-t megelőző munka első műveletének befejezési ideje), és R pedig a J_i - t megelőző munka második műveletének befejezési idejét.

Vegyük észre, hogy mivel S és S' csak a J_i és J_l munkák sorrendjében különbözik, ezért az állítás igazolásához elegendő belátni, hogy $C_l \geq C'_i$, ahol C_l a J_l munka befejezési ideje az S ütemezésben C'_i pedig a J_i munka befejezési ideje az S' ütemezésben.

Vizsgáljuk elsőként a C_l értéket. Az R és Q időpontok definíciója pontosan azt jelenti, hogy S-ben a J_i munka O_{i2} műveletének végrehajtása nem kezdődhet a $\max\{R, Q + \tau_{i1}\}$ időpont előtt. Másrészt $\tau_{l1} \leq \tau_{i2}$, tehát az O_{l1} művelet hamarább befejeződik az M_1 gépen, mint az O_{i2} művelet az M_2 gépen, így

$$C_l = \max\{R, Q + \tau_{i1}\} + \tau_{i2} + \tau_{l2}.$$

Most tekintsük a C_i' értéket. Az S' ütemezésben az O_{l2} művelet M_2-n a $\max\{R, Q+\tau_{l1}\}$ időpontban kezdődik, és az O_{i2} művelet pedig akkor, amikor O_{l2} és O_{i1} egyaránt befejeződött. Tehát

$$C_i' = \max\{\max\{R, Q+\tau_{l1}\}+\tau_{l2}, Q+\tau_{l1}+\tau_{i1}\}+\tau_{i2},$$

azaz

$$C_i' = \max\{\max\{R, Q+\tau_{l1}\}+\tau_{l2}+\tau_{i2}, Q+\tau_{l1}+\tau_{i1}+\tau_{i2}\}.$$

Másrészt $C_l \geq \max\{R, Q+\tau_{l1}\}+\tau_{l2}+\tau_{i2}$, mivel $\tau_{l1} \leq \tau_{i1}$. Továbbá $C_l \geq Q+\tau_{l1}+\tau_{i1}+\tau_{i2}$, mivel $\tau_{l1} \leq \tau_{l2}$.

Következésképp azt kaptuk, hogy C_l nem kisebb a C_i' kifejezésében szereplő maximum egyik tagjánál sem, így $C_i' \leq C_l$, amivel a tételt igazoltuk.

Mindenképpen érdemes megjegyeznünk, hogy az alábbi tétel alapján következik, hogy két gép esetén nem megszorítás, pusztán olyan ütemezéseket vizsgálni, amelyekben a π_j munkasorrendek megegyeznek minden gépre.

15.8. tétel. *Az $F2||C_{max}$ probléma esetén van olyan optimális ütemezés, amelyben a munkák sorrendje mindkét gépen ugyanaz.*

Bizonyítás. Tekintsük az optimális ütemezéseket. Minden optimális ütemezésre van olyan $0 \leq k \leq n$, hogy az első k munkának megegyezik a sorrendje a két gépen. Vegyünk egy olyan ütemezést, amelyre ez a k maximális. Tegyük fel, hogy erre az ütemezésre $k < n$. Legyen i a k-adik munka és j az a munka, amelyet az M_2 gépen közvetlenül az i munka második művelete után hajtunk végre.

Könnyen adódik, hogy amennyiben az első gépen j első műveletét közvetlenül i első művelete után hajtjuk végre és a többi, i után következő művelet kezdési idejét p_{1j}-vel növeljük, akkor egy olyan ütemezést kapunk, amelyben a maximális befejezési idő megegyezik az eredetivel, így szintén optimális. Másrészt ebben az új ütemezésben már az első $k+1$ munka sorrendje azonos a két gépen, ami ellentmond k maximalitásának. Következésképp a választott ütemezésben $k = m$, amivel igazoltuk a tétel állítását.

A Johnson-féle eljárás bemutatására tekintsük a következő feladatot.

15.3. példa. Tekintsünk egy olyan ütemezési feladatot, amelyben két gépünk van, M_1 és M_2, továbbá 6 munkát kell végrehajtani, a J_i, $(i = 1, \ldots 6)$ munkákat. A munkák mindegyike két műveletből áll, melyek közül

az elsőt M_1-en, a másodikat M_2-n kell végrehajtani. A munkákat és műveleteiket, valamint a műveletek végrehajtási idejeit adja meg az alábbi táblázat:

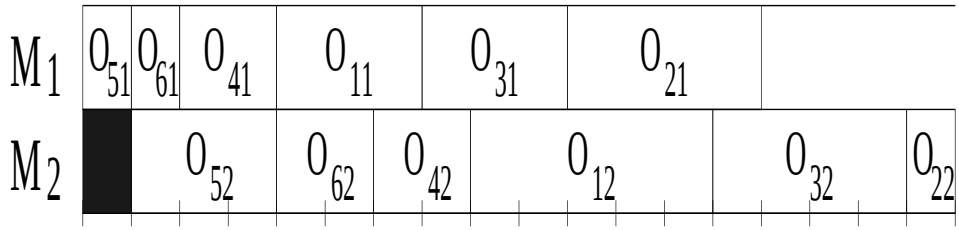
Munkák	Műveletek	Végrehajtási idők
J_1	O_{11}, O_{12}	3,5
J_2	O_{21}, O_{22}	4,1
J_3	O_{31}, O_{32}	3,4
J_4	O_{41}, O_{42}	2,2
J_5	O_{51}, O_{52}	1,3
J_6	O_{61}, O_{62}	1,2

Első lépésként beállítjuk a k és l értékeket. A $k := 1$ értékadás a 6-elemű lista elejére berakandó elem pozícióját, amíg az $l := 6$ értékadás a lista végére helyezendő elem helyzetét adja meg.

Képezve a végrehajtási idők minimumát az 1 értéket kapjuk, amely a 2 indexet határozza meg. Mivel $\tau_{22} = 1$, ezért a 4. lépésben a J_2 munka kerül a lista utolsó helyére, az l értéke 5-re változik, majd rátérünk ismét a 2. lépésre. A maradék munkák végrehajtási idejeinek minimuma ismét 1, ami most az 5 indexet határozza meg. Mivel $\tau_{51} = 1$, ezért J_5 kerül a lista első helyére és k értéke 2-re változik. A maradék munkákra a végrehajtási idők minimuma 1, ami most egyértelműen meghatározza a 6 indexet. Mivel $\tau_{61} = 1$, ezért J_6 kerül a lista második helyére, és k a 3 értéket veszi fel. A maradék munkákra a minimum 2, amely a 4 indexet határozza meg egyértelműen, és J_4 lesz a lista harmadik eleme, k pedig 4-re változik. Most a minimális elem 3, ami az 1 indexet határozza meg, és J_1 lesz a lista negyedik eleme, k pedig 5 értéket kap. Végül J_3 kerül a lista ötödik helyére, amivel az eljárás befejeződik. A listán levő elemek sorrendje a következő:

$$J_5 \prec J_6 \prec J_4 \prec J_1 \prec J_3 \prec J_2.$$

A megfelelő ütemezést, ami optimális megoldása a feladatnak, mutatja a 15.4. ábrán megadott Gannt diagramm.



15.4. ábra. A 15.3. példa optimális megoldása.

A fenti eljárást fejlesztette tovább J. R. Jackson [97] egy speciális job shop ütemezési modell megoldására. A továbbiakban olyan job shop ütemezési problémát tekintünk, amelyben két gép van M_1 és M_2, minden munka legfeljebb két műveletből áll, és amennyiben egy munka két műveletből áll akkor azt a két műveletet különböző gépeken kell végrehajtani, amelyek hozzá vannak rendelve a műveletekhez. Ezt a feladatot oldja meg a következő eljárás.

Jackson algoritmusa ([97])

- *1. lépés.* Képezzük a következő halmazokat:

  $K_{12} = \{$ azon két műveletből álló munkák, amelyek műveleteiből az elsőt az M_1, a másodikat az M_2 gépen kell végrehajtani $\}$,

  $K_{21} = \{$ azon két műveletből álló munkák, amelyek műveleteiből az elsőt az M_2, a másodikat az M_1 gépen kell végrehajtani $\}$,

  $K_1 = \{$ azon munkák, amelyek egy, az M_1-en végrehajtandó műveletből állnak $\}$,

  $K_2 = \{$ azon munkák, amelyek egy, az M_2-en végrehajtandó műveletből állnak $\}$.

- *2. lépés.* Rendezzük a K_{12} és a K_{21} halmazokat a Johnson eljárás szerint.

- *3. lépés.* Rendezzük a $K_{12} \cup K_1 \cup K_{21}$ halmaz elemeit úgy, hogy K_1-ben tetszőleges legyen a rendezés, és K_{12} legnagyobb eleme kisebb legyen mint K_1 legkisebb eleme, továbbá K_1 legnagyobb eleme kisebb legyen, mint K_{21} legkisebb eleme. Jelölje ezt a rendezést $(K_1', \prec)$.

- *4. lépés.* Rendezzük a $K_{21} \cup K_2 \cup K_{12}$ halmaz elemeit úgy, hogy K_2-ben tetszőleges legyen a rendezés, és K_{21} legnagyobb eleme kisebb legyen,

mint K_2 legkisebb eleme, továbbá K_2 legnagyobb eleme kisebb legyen, mint K_{12} legkisebb eleme. Jelölje ezt a rendezést $(K_2', \prec)$.

- *5. lépés.* Az M_1 gépen ütemezzük a munkákat $(K_1', \prec)$ szerint, az M_2-n pedig $(K_2', \prec)$ szerint.

15.9. tétel. *Az algoritmus valóban egy optimális megoldását adja meg a job shop problémának két gép esetén.*

Bizonyítás. Elsőként tegyük fel, hogy $\sum_{i \in K_{21}} p_{i2} \leq \sum_{i \in K_{12}} p_{i1} + \sum_{i \in K_1} p_{i1}$. Ebben az esetben az M_1 gépen nincs üres idő. Amennyiben az M_2 gépen sincs üres idő vagy a maximális befejezési idő $\sum_{i \in K_{12} \cup K_1 \cup K_{21}} p_{i1}$, akkor az ütemezés nyilvánvalóan optimális. Ellenkező esetben, a maximális befejezési idő pontosan megegyezik a K_{12} halmazra megszorított probléma optimális ütemezésében a befejezési idővel, azaz ekkor is optimális megoldást kaptunk. Amennyiben $\sum_{i \in K_{21}} p_{i2} > \sum_{i \in K_{12}} p_{i1} + \sum_{i \in K_1} p_{i1}$, akkor az M_2 gépen nincs üres idő, és az algoritmus által kapott ütemezés optimalitása az előző esethez teljesen hasonlóan látható.

A Jackson-féle eljárás bemutatására oldjuk meg a következő ütemezési feladatot.

15.4. példa. Tegyük fel, hogy két gépünk van, M_1 és M_2, továbbá 8 munkát kell végrehajtani, a J_i, $(i = 1, \ldots, 8)$ munkákat. A munkák mindegyike legfeljebb két műveletből áll, és a J_i munkára az $m_{i1} \neq m_{i2} \in \{1, 2\}$ paraméterek adják meg, hogy az O_{i1} és O_{i2} műveleteket melyik gépen kell végrehajtani. A munkákat és műveleteiket, valamint a műveletek végrehajtási idejeit és a gépeket meghatározó paramétereket tartalmazza az alábbi táblázat:

Munkák	Műveletek	Végrehajtási idők	m_{i1}, m_{i2}
J_1	O_{11}, O_{12}	1,1	1,2
J_2	O_{21}, O_{22}	2,1	2,1
J_3	$O_{31}, -$	3,-	1,-
J_4	O_{41}, O_{42}	1,4	1,2
J_5	O_{51}, O_{52}	2,3	2,1
J_6	$O_{61}, -$	2,-	2,-
J_7	$O_{71}, -$	1,-	1,-
J_8	$O_{81}, -$	2,-	2,-

Ekkor $K_{12} = \{J_1, J_4\}$, $K_{21} = \{J_2, J_5\}$, $K_1 = \{J_3, J_7\}$ és $K_2 = \{J_6, J_8\}$. Rendezve a Johnson-féle algoritmus szerint a K_{12} és K_{21} halmazokat, az alábbi rendezéseket kapjuk:

$$J_1 \prec J_4, \quad J_5 \prec J_2.$$

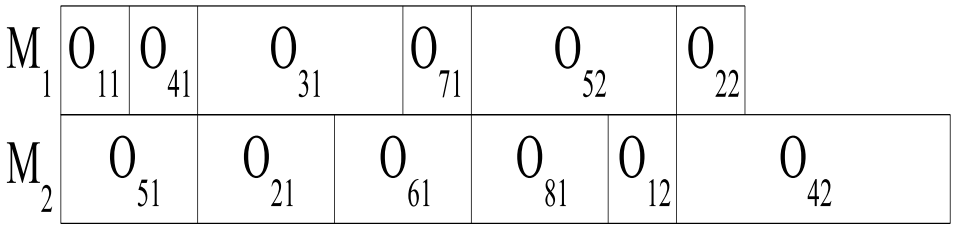
Most a szükséges halmazok rendezése a következő lesz:

$$K_{12} \cup K_1 \cup K_{21} : \quad J_1 \prec J_4 \prec J_3 \prec J_7 \prec J_5 \prec J_2,$$

$$K_{21} \cup K_2 \cup K_{12} : \quad J_5 \prec J_2 \prec J_6 \prec J_8 \prec J_1 \prec J_4.$$

A halmazok rendezésének megfelelő ütemezés lesz a probléma optimális megoldása.

A megfelelő ütemezést, ami optimális megoldása a feladatnak, mutatja a 15.5. ábrán megadott Gannt diagramm.



15.5. ábra. A 15.4. példa optimális megoldása.

16. Approximációs sémák

Az approximációs sémák tulajdonképpen heurisztikus algoritmusokból álló algoritmusosztályok. Heurisztikus algoritmusoknak egy olyan sorozatát keressük, amely tetszőlegesen kicsi ε esetén tartalmaz $1+\varepsilon$-approximációs algoritmust. Minden algoritmusnak polinomiális idejűnek kell lennie, de az időigény kiszámítása során az ε értéket konstansként kezeljük. Az időigény és az ε érték kapcsolatától függően az alábbi sémákat definiálhatjuk.

Algoritmusok egy H_ε osztályát *polinomiális idejű approximációs sémának* nevezzük (polynomial time approximation scheme, PTAS rövidítve), ha minden $\varepsilon > 0$ esetén H_ε egy $1+\varepsilon$-approximációs algoritmus, amely polinomiális időigényű, ahol az időigény tetszőlegesen függhet a ε konstanstól. Amennyiben egy H_ε polinomiális idejű approximációs sémára, a H_ε algoritmusok időigénye nem csak az input méretében, hanem $1/\varepsilon$-ban is polinomiális az algoritmusosztályt *teljesen polinomiális approximációs sémának* nevezzük (fully polynomial time approximation scheme, FPTAS rövidítve).

Az approximációs sémák területe igen széles körben vizsgált. Ebben a fejezetben néhány példán keresztül bemutatjuk az alapvető technikákat, amelyeket ütemezési problémák esetén approximációs sémák kidolgozására alkalmazni szoktak. A könyvben tárgyalt egyéb problémákkal részletesen nem foglalkozunk. A hátizsák probléma esetére FPTAS található a [72] és [94] dolgozatokban, és egy gyorsabb FPTAS a [108] munkában. A halmazlefedési és a TSP problémák esetén a $P \neq NP$ feltétel mellett nem létezik konstans approximációs hányadossal rendelkező heurisztikus algoritmus, így nem ismertek approximációs sémák sem ezen problémákra. Fontosnak tartjuk megjegyezni, hogy olyan TSP feladat esetén, amelyben a távolságmátrix euklideszi (a városok egy euklideszi tér pontjai és a távolságmátrixot az euklideszi metrika alapján kapjuk), ismert FPTAS a probléma megoldására, az algoritmusok megtalálhatók a [4] és [143] dolgozatokban. A kiszolgálási feladatokról részletek találhatóak az [5] és [93] cikkekben illetve az ezekben található további hivatkozásokban. Az érdeklődő olvasó az approximációs sémák területének részletes bemutatását megtalálhatja a [160], [172] munkákban.

Approximációs sémák esetén polinomiális idejű algoritmussal szeretnénk kapni egy megoldást. Lényegében az eredeti problémát egyszerűsítjük le úgy, hogy egyrészt az egyszerűsített változat már polinomiális időben megold-

ható legyen, másrészt ne veszítsünk sokat a megoldás pontosságából. Az approximációs algoritmusok általában három osztályba sorolhatóak aszerint, hogy a probléma egyszerűsítését hol hajtjuk végre. Egyszerűsíthetünk, az inputon, az outputon és a megoldást adó algoritmus egyes fázisai között. Elsőként az egyik legegyszerűbb NP-nehéz ütemezési problémára (amelyben két azonos gép van és a maximális befejezési időt minimalizáljuk) mutatunk be három approximációs sémát, mindhárom, a fentiekben említett megközelítésre egyet - egyet, majd egy bonyolultabb eljárást ismertetünk, amely segítségével dinamikus programozási algoritmusokat transzformálhatunk approximációs sémává.

Az alábbi három példában jelölje $J = \{J_1, \ldots, J_m\}$ a munkák halmazát, rendre $p_1, \ldots, p_m$ a végrehajtási időket, és legyen $\varepsilon > 0$ tetszőleges kicsi szám. Legyen továbbá $L = \max\{\max p_i, (\sum_{i=1}^{m} p_i)/2\}$. Ekkor miként azt a 15.5. tétel bizonyítása során igazoltuk, két gép esetén $L \leq OPT(J) \leq 2L$.

Az első séma alapgondolata az, hogy az apró munkákat nem kezeljük egyenként, hanem olyan tölteléknek tekintjük őket, amelyeknek csak az összköltése számít.

Inputban egyszerűsítő séma (IES_ε)

1. lépés. Osszuk a munkákat két halmazba. Legyenek az $\varepsilon L/2$-nél nem kisebb munkák a nagy munkák, a kisebbek a kicsi munkák. Konstruáljuk a következő J' inputot. A nagy munkákat változatlanul hagyjuk, a kicsi munkákat pedig $\varepsilon L/2$ nagyságú munkákkal helyettesítjük a következőképpen. Ragasszuk össze az összes kicsi munkát egyetlen óriásmunkába, és az így kapott óriásmunkát szeleteljük fel $\varepsilon L/2$ nagyságú darabokra, a maradék $\varepsilon L/2$-nél kisebb darabot dobjuk el. Nevezzük ezeket a munkákat gyűjtőmunkáknak.

2. lépés. Oldjuk meg a J' inputra a problémát, az összes lehetséges megoldást megvizsgálva. Mivel az egyes gépeken nem számít a munkák sorrendje, ezért feltehetjük, hogy a kapott optimális megoldásban mindkét gépen először a régi nagy munkákat hajtjuk végre, utána a gyűjtőmunkákat. Továbbá a gyűjtőmunkák mind $\varepsilon L/2$ nagyságú munkák, így feltehetjük, hogy az első gépen a gyűjtőmunkák a felszeletelt óriásmunka kezdőszeletét alkotják, a második gépen a végszeletét. Jelölje ezt az ütemezést S'.

3. lépés. Az S' ütemzésből konstruáljuk meg J egy ütemezését. A nagy munkákat ütemezzük ugyanazon a gépen, mint S', a gyűjtőmunkákat pedig bontsuk vissza az eredeti kicsi munkákra, továbbá az óriásmunka konstruálásánál eldobott részt még illesszük a második gép végére. Mivel a gépeken

az óriásmunka kezdő illetve végszelete van, ezért a visszabontás során csak az első gép utolsó, és a második gép első gyűjtőmunkájánál keletkezik nem visszabontható, szétvágott kicsi munka. Ezt ütemezzük az első gépen.

16.1. tétel. *Az IES_ε séma egy PTAS.*

Bizonyítás. Vizsgáljuk elsőként a futási időt! Az első és a harmadik lépés nyilvánvalóan polinomiális. A második lépésben a J' inputra megvizsgáljuk az összes lehetséges megoldást, ez $O(2^{m'})$ lépés, ahol m' a munkák száma. Másrészt minden munka legalább $\varepsilon L/2$ méretű, a munkák végrehajtási idejeinek összege az átdarabolás során nem növekedhetett, így a munkák száma legfeljebb $4/\varepsilon$. Következésképp rögzített konstans ε mellett ezen rész futási ideje konstans. (Érdemes megjegyeznünk, hogy ez a konstans exponenciális $1/\varepsilon$-ban, ezért nem FPTAS-t kapunk.) Következésképp a futási idő valóban polinomiális az input méretében minden ε-ra.

Most vizsgáljuk az approximációs hányadost! Elsőként vegyük észre, hogy $OPT(J') \leq OPT(J) + \varepsilon L/2$. Valóban J egy optimális ütemezésében, a nagy munkákat helyben hagyva, a kicsi munkákat kicserélve annyi $\varepsilon L/2$ nagyságú munkával, hogy az össztöltésük ne csökkenjen, egy olyan lehetséges megoldását kapjuk J'-nek, amelyre a maximális befejezési idő legfeljebb $\varepsilon L/2$-vel növekedett. Másrészt a gyűjtőmunkák szétdarabolása során is legfeljebb $\varepsilon L/2$-el növekszik a maximális befejezési idő, hiszen az első gépen az utolsó kicsi munka másik gyűjtőmunkába eső darabja, a második gépen az óriásmunkából eldobott rész növelheti meg a maximális befejezési időt.

Következésképpen

$$IES_\varepsilon(J) \leq OPT(J') + \varepsilon L/2 \leq OPT(J) + 2\varepsilon L/2 \leq (1+\varepsilon)OPT(J),$$

amivel igazoltuk, hogy a definiált eljárásosztály valóban egy approximációs sémát ad.

Az outputban egyszerűsítő sémák alapgondolata általában az, hogy a lehetséges megoldásokat polinomiális számú osztályba osztjuk fel, és minden osztályra alkalmazunk egy gyors algoritmust, amely kijelöl az osztályból egy reprezentánst. Amennyiben minden reprezentáns közel van az osztályba tartozó legjobb lehetséges megoldáshoz, akkor a reprezentánsok közül a legjobb közel lesz az optimális megoldáshoz, így egy jó approximációs algoritmust kapunk.

Outputban egyszerűsítő séma (OES_ε)

1. lépés. Osszuk a munkákat két halmazba. Legyenek az εL-nél nem kisebb munkák a nagy munkák, a kisebbek a kicsi munkák. Vegyük a nagy munkák összes lehetséges ütemezését, ezeket nevezzük parciális ütemezéseknek. Minden parciális ütemezésre definiáljuk a lehetséges ütemezéseknek egy osztályát, amely azokat a lehetséges megoldásokat tartalmazza, amelyekben a nagy munkák az adott parciális ütemezésnek megfelelően vannak a gépekhez hozzárendelve.

2. lépés. Minden osztályra számoljunk ki egy reprezentánst. A reprezentánst úgy kapjuk, hogy a parciális ütemezést kiterjesztjük a kicsi munkákkal, azokat a LISTA algoritmus szerint ütemezve (az aktuális munka arra a gépre kerül, ahol kisebb az aktuális töltés).

3. lépés. Vegyük a reprezentánsok közül a legkisebb célfüggvényértékkel rendelkező megoldást.

16.2. tétel. *Az OES_ε séma egy PTAS.*

Bizonyítás. Vizsgáljuk elsőként a futási időt! Mivel a nagy munkák mérete legalább εL és a munkák össztöltése legfeljebb $2L$, ezért a nagy munkák száma legfeljebb $2/\varepsilon$. Következésképpen a parciális ütemezések, és így az osztályok száma legfeljebb $2^{2/\varepsilon}$. Minden osztályra a reprezentáns kijelölése lineáris időben fut, így rögzített ε mellett az algoritmus futási ideje valóban polinomiális. (Fontos megjegyeznünk, hogy $1/\varepsilon$-ban exponenciális, így ismét csak PTAS-t kapunk.)

Vizsgáljuk most az approximációs hányadost! Különböztessünk meg két esetet attól függően, hogy milyen reprezentánsokat kaptunk!

Elsőként tegyük fel, hogy van olyan reprezentáns, amelyben egy kicsi munka éri el a maximális befejezési időt. Ez akkor fordul elő, ha mindkét gépre kerül kicsi munka vagy, ha csak a parciális ütemezésben kisebb töltésűre, de az utolsó kicsi munkával a töltés túlnő a parciális ütemezésben nagyobb töltésű gép töltésén. Mindkét esetben ezen reprezentánsra a gépeken a befejezési idők (esetünkben megegyeznek a töltéssel) különbsége legfeljebb εL. Másrészt az össztöltés legfeljebb $2L$, így azt kaptuk, hogy

$$OES_\varepsilon(J) \leq L + \frac{\varepsilon}{2}L \leq (1 + \varepsilon/2)OPT(J).$$

A második esetben nincs olyan reprezentáns, amelyben kicsi munka éri el a maximális befejezési időt. Ez azt jelenti, hogy minden osztályra a reprezentáns optimális megoldást ad, hiszen a költség megegyezik a parciális

ütemezés költségével. Másrészt ebből következik, hogy a reprezentánsok között szerepel egy optimális megoldás is.

A harmadik megközelítést akkor szokás használni, ha van a problémára egy exponenciális megoldó algoritmusunk, amely több fázisból áll. Ekkor bizonyos esetekben, az egyes fázisok közé valamilyen egyszerűsítő lépést beiktatva, egy approximációs sémát kaphatunk.

Elsőként tekintsük a következő egyszerű algoritmust, amely meghatározza az összes (a, b) párt, amelyek előállhatnak töltésként a két gépen a J munkahalmaz ütemezése után!

Egy exponenciális idejű algoritmus

1. lépés. Legyen $i = 0$, $S_0 = \emptyset$.

2. lépés. Amennyiben $i = m$ véget ért az eljárás, egyébként minden $(a, b) \in S_i$ párra képezzük az $(a + p_{i+1}, b)$ és $(a, b + p_{i+1})$ párokat, és ezen párok alkossák az S_{i+1} halmazt. Növeljük i értékét 1-gyel és ismételjük meg a 2. lépést.

Teljes indukcióval egyszerűen igazolható, hogy az S_i halmaz minden i-re azokat az (a, b) párokat tartalmazza, amelyek előállhatnak töltésként a két gépen a J munkahalmaz első i elemének az ütemezése után. Következésképp kiválasztva az S_m halmazból azt a párt, amelyre az elemek maximuma a minimális, megkapjuk az optimális célfüggvényértéket. Amennyiben a párokat kiegészítjük azzal (a dinamikus programozási algoritmusok esetén szokásos) információval, hogy miként kaptuk meg a párt a megelőzőből (ez egy extra bit, amely megadja, hogy az utolsó munkát melyik géphez rendeltük), akkor megkapjuk az optimális megoldást is.

Tehát a fenti algoritmus alapján megkaphatjuk az optimális ütemezést. Az egyetlen probléma az, hogy az S_i halmazok elemszáma rendkívül gyorsan növekedhet, elérheti a 2^i értéket. Amennyiben ezt az elemszámot korlátozni tudjuk, akkor egy polinomiális algoritmust kapunk. Az alábbiakban megvizsgáljuk miként tehetjük ezt meg oly módon, hogy ne veszítsünk sokat a pontosságból.

Legyen $\Delta = 1 + \varepsilon/(2m)$ és $P = \sum_{i=1}^{m} p_i$. Osszuk fel a $P \times P$ nagyságú négyzetet tartományokra! A P_{jk}, $j \geq 1$, $k \geq 1$ tartomány tartalmazza azokat az (a, b) pontokat, amelyekre $\Delta^j \leq a \leq \Delta^{j+1}$, $\Delta^k \leq b \leq \Delta^{k+1}$ teljesül. A P_{0k}, $k \geq 1$ tartomány tartalmazza azokat az (a, b) pontokat, amelyekre $0 \leq a \leq \Delta$, $\Delta^k \leq b \leq \Delta^{k+1}$ teljesül. A P_{j0}, $j \geq 1$ tartomány tartalmazza azokat az (a, b) pontokat, amelyekre $\Delta^j \leq a \leq \Delta^{j+1}$, $0 \leq b \leq \Delta$ teljesül. A P_{00} tartomány tartalmazza azokat az (a, b) pontokat, amelyekre $0 \leq a \leq \Delta$, $0 \leq b \leq \Delta$ teljesül. Az approximációs séma alapgondolata, hogy

az ugyanazon tartományba eső pontpárok közel vannak egymáshoz, így elég mindegyik tartományból egy pontot megőrizni az S_i halmazokban.

Algoritmusban egyszerűsítő séma (AES_ε)

1. lépés. Legyen $i = 0$, $S_0 = \emptyset$.

2. lépés. Amennyiben $i = m$, akkor véget ért az eljárás. Egyébként minden $(a, b) \in S_i$ párra képezzük az $(a + p_{i+1}, b)$ és $(a, b + p_{i+1})$ párokat, és ezen párok alkossák az S'_{i+1} halmazt.

3. lépés (ritkítás). Az S'_{i+1} azon elemeinek halmazát, amelyben egyik érték sem 0 ritkítsuk úgy, hogy minden P_{jk} tartományból csak egy elemet tartsunk meg. Legyen S_{i+1} a ritkított S'_{i+1}. Növeljük i értékét 1-gyel és folytassuk az eljárást a 2. lépéssel.

16.3. tétel. *Az AES_ε séma egy FPTAS.*

Bizonyítás. Vizsgáljuk elsőként a futási időt! Ehhez össze kell számolnunk a tartományokat. A definícióból egyből adódik, hogy a tartományok száma legfeljebb $(\log_\Delta P + 1)^2 = O((\log_\Delta P)^2)$. Másrészt

$$\log_\Delta P = \frac{\ln P}{\ln \Delta}.$$

Továbbá $\ln\Delta = \ln(1 + \varepsilon/(2m)) = \Omega(\varepsilon/m)$, amiből adódik, hogy a tartományok száma polinomiális m-ben $1/\varepsilon$-ban és $\ln P$-ben, azaz az input méretében és $1/\varepsilon$-ban. Ebből következik, hogy az algoritmus is polinomiális ezekben az értékekben, azaz futási idő szempontjából valóban egy FPTAS-t adtunk meg.

Vizsgáljuk az approximációs hányadost! Minden iterációban van egy ritkító fázis. A ritkító részben minden párhoz egy olyan párt őrzünk meg, amely ugyanabban a tartományban van, így mindkét komponensben a megfelelő érték legfeljebb egy multiplikatív Δ hibában tér el. Következésképp minden lehetséges ütemezést leíró párhoz van az S_m halmazban egy olyan pár, amely legfeljebb egy Δ^m multiplikatív hibában tér el. Tehát az algoritmus Δ^m-approximációs. Másrészt egyszerűen adódik az analízisben ismert $0 \le x \le 1$ számokra fennálló $(1 + x/m)^m \le 1 + 2x$ egyenlőtlenség alapján, hogy

$$\Delta^m = (1 + \varepsilon/2m)^m \le 1 + \varepsilon,$$

amivel az állítást igazoltuk.

A fenti eljárás könnyen módosítható úgy, hogy az optimális ütemezést is megkapjuk. Számhármasokat kell használnunk, ahol a harmadik komponens

attól függ, hogy az utolsó munkát melyik gépen ütemeztük a töltéseket eredményező ütemezésben. Ebben az esetben az optimális (approximációs) számhármasból visszafejthető egy optimális (approximációs) ütemezés.

A fejezet utolsó részében arra mutatunk példát, hogy miként transzformálhatóak bizonyos dinamikus programozási algoritmusok FPTAS sémává. Az FPTAS alapötlete az, hogy az inputot kerekítjük, az így kapott új inputra végrehajtjuk a dinamikus programozási eljárást, amely a kerekített inputra polinomiális idejű lesz, és a kapott optimális megoldást visszatranszformáljuk az eredeti feladat egy lehetséges megoldásává. Tehát az alábbi algoritmus is az inputban egyszerűsítő algoritmusok családjába tartozik. A bemutatott algoritmus lényegében megegyezik a [91] cikkben ismertetett eljárással. A továbbiakban az eddig vizsgált alapproblémát n gép esetén vizsgáljuk. Feltesszük, hogy a végrehajtási idők egészek, így a befejezési időkről is feltehetjük, hogy egészek. Elsőként egy dinamikus programozási eljárást ismertetünk a feladat megoldására.

Legyen $P = \sum_{j=1}^{m} p_j$. Definiáljuk az $S_i(K_1, \ldots, K_{n-1})$ értékeket, $(i = 0, 1, \ldots, m)$ a következőképpen. Legyen $S_i(K_1, \ldots, K_{n-1})$ az a minimális töltés, amely elérhető az M_n gépen az első i munka ütemezése során olyan ütemezéssel, amelyben az M_j gépen a töltés legfeljebb K_j minden $j = 1, \ldots, n-1$-re. A függvény definíciója alapján $S_i(K_1, \ldots, K_{n-1}) = \infty$, ha $K_j < 0$ valamely j-re, hiszen ekkor nincs a j-edik gép töltésére vonatkozó feltételt kielégítő ütemezés. Továbbá $S_0(K_1, \ldots, K_{n-1}) = 0$ minden $0 \leq K_j \leq P$ értékre.

Egyszerű esetszétválasztás alapján adódik, hogy az S_i függvényekre teljesül a következő rekurzió:

$$S_{i+1}(K_1, \ldots, K_{n-1}) = \min\{p_{i+1} + S_i(K_1, \ldots, K_{n-1}), \mathrm{MIN}\},$$

ahol

$$\mathrm{MIN} = \min_j\{S_i(K_1, \ldots, K_{j-1}, K_j - p_{i+1}, K_{j+1}, \ldots, K_{n-1}\}.$$

A fenti észrevételek alapján könnyű felépíteni egy dinamikus programozási algoritmust, amely megoldja a $Pn||C_{max}$ problémát egész végrehajtási idők mellett. A kezdeti feltételek és a rekurzió alapján kiszámoljuk az $S_m(K_1, \ldots, K_{n-1})$ értékeket, minden egész $0 \leq K_j \leq P$, $j = 1, \ldots, n-1$ értékre. Ezt követően az optimális célfüggvényérték a minimális

$$\max\{S_m(K_1, \ldots, K_{n-1}), \max_{1 \leq j \leq n-1} K_j\}$$

érték lesz. Másrészt, ha az eljárás során (amint ez a dinamikus programozási algoritmusoknál szokásos) mindig megjegyezzük, hogy az aktuális S_{i+1} függvényértéket melyik S_i érték alapján kapjuk, akkor az optimális célfüggvényértéket adó S_m érték alapján visszafejthető az optimális ütemezés.

Tehát a fentiekben bemutatott algoritmus megoldja a problémát. Vizsgáljuk a futási időt. Az eljárás során lényegében ki kell töltenünk m darab P^{n-1} méretű táblázatot (a lehetséges $K_1, \ldots, K_{n-1}$ értékek), minden érték kiszámítása n műveletet igényel, tehát a futási idő $O(mnP^{n-1})$, azaz rögzített n mellett polinomiális m-ben és $P = \sum_{j=1}^{m} p_j$-ben. A probléma az, hogy az input mérete nem P, hanem $\log P$, mivel binárisan reprezentáljuk a számokat. Unáris reprezentálás mellett az algoritmus polinomiális lenne. (Érdemes megjegyeznünk, hogy az ilyen algoritmusokat *pszeudopolinomiális* algoritmusoknak nevezzük.) Az alábbiakban bemutatjuk miként konstruálható egy FPTAS a fentiekben bemutatott dinamikus programozási eljárás alapján. A módszer végrehajthatóságának feltételeit vizsgálja a [177] dolgozat.

Az FPTAS előkészítő részeként hajtsunk végre egy konstans approximációs hányadossal rendelkező approximációs eljárást az inputon. A kapott célfüggvényértéket jelölje L. Esetünkben használhatjuk a LISTA algoritmust, amely lineáris időigényű. Ekkor az L számra teljesül az $OPT(J) \leq L \leq 2OPT(J)$ egyenlőtlenség.

Dinamikus programozási séma (DPS_ε)

1. lépés. Konstruáljuk meg a J inputból a J' inputot a következőképpen. Minden j-re a p_j végrehajtási idővel rendelkező j-edik munkát helyettesítsük egy $p'_j = \lfloor p_j/Z \rfloor$ végrehajtási idővel rendelkező munkával, ahol $Z = \varepsilon L/2m$.

2. lépés. Oldjuk meg az 1. lépés során kapott J' inputot a fentiekben ismertetett dinamikus programozási eljárással.

3. lépés. A J' input optimális ütemezéséből J egy közelítő megoldását kapjuk, ha minden munkát ugyanazon a gépen ütemezünk, amelyen a módosított input optimális megoldása a megfelelő módosított munkát ütemezi.

16.4. tétel *A DPS_ε séma egy FPTAS.*

Bizonyítás. Elsőként vizsgáljuk a futási időt! A dinamikus programozási algoritmus időigénye $O(mnP^{n-1})$. A módosított J' inputra

$$P' \leq \sum_{j=1}^{m} \lfloor p_j/Z \rfloor \leq \sum_{j=1}^{m} p_j/Z \leq nOPT(J)/Z = \frac{nOPT(J)}{\varepsilon L/2m} \leq 4mn/\varepsilon.$$

Következésképp az eljárás időigénye rögzített n konstans mellett valóban polinomiális az input méretében és $1/\varepsilon$-ban.

Vizsgáljuk meg az approximációs hányadost!

Mivel $Z \cdot OPT(J') \leq OPT(J)$ és $DPS_\varepsilon(J) \leq Z \cdot OPT(J') + mZ$, ezért

$$|DPS_\varepsilon(J) - OPT(J)| \leq mZ.$$

Következésképpen:

$$\frac{|DPS_\varepsilon(J) - OPT(J)|}{OPT(J)} \leq \frac{mZ}{OPT(J)} \leq \frac{\varepsilon L}{2OPT(J)} \leq \varepsilon.$$

Irodalomjegyzék

[1] Adrabinski, A., M. M. Syslo, Computational experiments with some heuristic algorithms for the traveling salesman problem, TR. N-78, Institute of Computer Science, Wroclaw University, Wroclaw, Poland, 1980; also in *Zatos. Mat.* **18** (1983), 91-95.

[2] Aho, A. V., J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass, 1974.

[3] Andrásfai, B., *Gráfelmélet*, POLYGON Kiadó, Szeged, 1994.

[4] Arora, S., Polinomial time approximation schemes for Euclidean traveling salesman and other geometric problems, *Journal of the ACM* **45** (1998), 753-782.

[5] Arora, S, P. Raghavan, S. Rao, Approximation schemes for Euclidean k-medians and related problems. Proceedings of STOC '98 (ACM, New York), (1999) 106-113.

[6] Azevedo, J., J. Madeira, E. Martins, F. Pires, A Shortest Path Ranking Algorithm, in: *Proc. AIRO'90 - Models and Methods for Decision Support,* Sorrento (1990), 1001-1011.

[7] Bajalinov, E., Imreh, B., *Operációkutatás*, POLYGON Kiadó, Szeged, 2001.

[8] Baker, K. R., *Introduction to Sequencing and Scheduling*, Wiley, New York, 1974.

[9] Bakó, A., Marton, L., Takács, B., Vesztergál, L., Városi úthálózat változatok gazdaságossági elemzése, *Városi Közlekedés* **5** (1981), 245-262.

[10] Bakó, A., Marton, L., Takács, B., Vesztergál, L., Egy interaktív modell úthálózatok optimalizálására, *KTMF Tudományos Közlemények* **1** (1981), 33-36.

[11] Balas, E., E. Zemei, An Algorithm for Large Zero-One Knapsack Problems, *Operations Research* **28** (1980), 1132-1154.

[12] Balinski, M. L., Quandt, R. E., On an integer program for a delivery problem, *Operations Research* **12** (1964), 300-304.

[13] Bartalos, I., T. Dudás, B. Imreh, On a tour construction heuristic for the asymmetric TSP, *Acta Cybernetica* **12** (1995), 209-216.

[14] Bellman, R., Some Applications of the Theory of Dynamic Programming – A Review, *Operations Research* **2** (1954), 275-288.

[15] Bellman, R., Notes on the Theory of Dynamic Programming IV – Maximization Over Discrete Sets, Naval Research Logistics Quarterly **3** (1956), 67-70.

[16] Bellman, R., Comment on Dantzig's Paper on Discrete-Variable Extremum Problems, *Operations Research* **5** (1957), 723-724.

[17] Bellman, R., *Dynamic Programming*, Princeton University Press, 1957.

[18] Bellman, R. E., On a routing problem, *Quart. Appl. Math.* **16** (1958), 87-90.

[19] Bellmore, M., S. Goldwasser, C. Lund, A. Russel, Efficient probabilistically checkable proofs and applications to approximation, Proc. of 25-th STOC, 1993, 294-304.

[20] Bellmore, M., J. C. Malone, Pathology of traveling salesman subtour elimination algorithms, *Operations Research* **19** (1971), 278-307.

[21] Berge, C., Two theorems in graph theory, *Proceedings of the National Academy of Science of the U.S.* **43** (1957), 842-844.

[22] Bertier, P. and B. Roy, Procédure de résolution pour une classe de problems pouvant avoir un caractére combinatorie, *Cahiers Cent. d'Etudes Recherche Operationelle* **6** (1964), 202-208.

[23] Borgulya, I., *Evolúciós Algoritmusok*, Dialóg Campus Kiadó, 2004.

[24] Branco, I. M., J. D. Coelho, The Hamiltonian p-median problem, *European Journal of Operational Research* **47** (1990), 86-95.

[25] Bruckner, P., *Scheduling algorithms*, Springer, Berlin, 1998.

[26] Burkard, R. E., V. G., Deĭneko, R. van Dal, J. van der Veen and G. J. Woeginger, Well-Solvable Special Cases of the TSP: A Survey, *SIAM Review* **40**, (1998), 496-546.

[27] Burkard, R. E. and J. Krarup, A linear algorithm for the pos/neg- weighted 1-median problem on a cactus, *Computing* **60** (1998), 193–215.

[28] Burkard, R. E., J. Offermann, Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme, *Zeitschrift für Operations Research* **21** (1977), B121-B132.

[29] Busacker, R. G., P. J. Gowen, A procedure for determining a family of minimal cost network flow patterns, O. R. O. Technical Report **15** (1961).

[30] Carpaneto, G., P. Toth, Some new branching and bounding criteria for the asymmetric traveling salesman problem, *Management Sci.* **26** (1980), 736-743.

[31] Cerdeira, J. O., The Hamiltonian k-median problem for any given k is NP-complete, *Centro de Estatística e Aplicações*, Universidade de Lisboa, nota no. **18/85** (1986).

[32] Chen, B., C. N. Potts, G. J. Woeginger, A review of machine scheduling: Complexity, algorithms and approximability, *Handbook of Combinatorial Optimization Volume 3* eds.: D. Y. Du, P. Pardalos, Kluwer Academic publisher, 1998, 21–171

[33] Chen, B., C. N. Potts, G. J. Woeginger, A review of machine scheduling: Complexity, algorithms and approximability, Report, TU-Graz, 1998.

[34] Christofides, N. Worst-case analysis of a new heuristic for the traveling salesman problem, Technical Report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh PA, 1976.

[35] Christofides, N., S. Elion, Algorithms for Large-Scale Traveling Salesman Problems, *Operational Res. Quart.* **23** (1972), 511-518.

[36] Chvatal, V., A greedy heuristic for the set covering problem, *Math. Operations Research* **4** (1979), 233-235.

[37] Conway, R. W., W. L. Maxwell, L. W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading MA., 1967.

[38] Cook, W. J., W. H. Cunningham, W. R. Pulleyblank, A. Schrijver, *Combinatorial Optimization*, Jonn Wiley & Sons Inc., New York, Chichester, Weinheim, Brisbane, Singapore, Toronto, 1998.

[39] Dakin, R., A tree search algorithm for mixed integer programming problems, *Computer Journal* **8** (1965), 250-255.

[40] Dantzig, G. B., Discrete-Variable Extremum Problems, *Operations Research* **5** (1957), 266-277.

[41] Day, R. H., An optimal extracting form a multiple file datastorage system: An application of integer programming, *Operationals Research* **13** (1965), 428-494.

[42] Deo N., C. Pang, Shortest-Path Algorithms: Taxonomy and Annotation, *Networks* **14** (1984), 275-323.

[43] Dickey, J. W. and J. W. Hopkins, Campus building arrangement using TOPAZ, *Transportation Research* **6** (1972), 59-68.

[44] Dijkstra, E. W., A note on two problems in connection with graphs, *Numerische Mathematik* **1** (1959), 269-271.

[45] Du, D. Z., P. M. Pardalos (Eds.), *Handbook of Combinatorial Optimization*, Kluwer Academic Publishers, Dordrecht-Boston-London, 1998.

[46] Eastman, W. L., *Linear Programming with Pattern Constraints*, Ph.D. thesis, Harvard University, Cambridge, 1958.

[47] Edmonds, J., Paths, trees, and flowers, *Canadian Journal of Mathematics* **17** (1965), 449-467.

[48] Edmonds, J., E. Johnson, Matching: A Well Solved Class of Integer Linear Programs, *Combinatorial Structures and Their Applications*, Gordon and Breach, New York, 1970, 89-92.

[49] Edmonds, J., R. M. Karp, Theoretical improvements in algorithm efficiency for network flow problems, *J. ACM* **19** (1972), 218-264.

[50] Egerváry, J., Mátrixok kombinatorikus tulajdonságairól, *Mat. és Fiz. Lapok* **38** (1931), 16-28.

[51] Egerváry, J., On combinatorial properties of matrices, translated by H. W. Kuhn, *Logistics Papers* **11**, George Washington University, 1-11.

[52] Elshafei, A. N., Hospital layout as a quadratic assignment problem, *Operations Research Quarterly* **28** (1977), 167-179.

[53] Euler, L., Solutio problematis ad geometriam situs pertinentis, *Commentarii Academiae Scientarum Imperialis Petropolitanae* **8** (1736), 128-140.

[54] Evans, J. R., E. Minieka, *Optimization Algorithms for Networks and Graphs*, Marcel Dekker Inc., New York- Basel-Hong Kong, 1992.

[55] Fayard, D., G. Plateanu, Resolution of the $0-1$ Knapsack Problem: Comparison of Methods, *Mathematical Programming* **8** (1975), 272-307.

[56] Floyd, R. W., Algorithm 97, Shortest path, *Commun. ACM* **5** (1962), 345.

[57] Ford, L. R., *Network Flow Theory*, The Rand Corp., P923, 1955.

[58] Ford, L. R., D. R. Fulkerson, Maximal flow through a network, *Canadian Journal of Mathematics* **8** (1956), 399-404.

[59] Ford, L. R., D. R. Fulkerson, A simple algorithm for finding maximal network flow and an application to the Hitchcock problem, *Canadian J. Math.* **9** (1957), 210-218.

[60] Ford. L. R., D. R. Fulkerson, *Flows in Networks*, Princeton University Press, Princeton, New Jersey, 1962.

[61] Frederickson, G. N., D. B. Johnson, Finding k-th paths and p-centers by generating and searching good data structures, *Journal of Algorithms* **4** (1983), 61-80.

[62] Fredman, M. L., R. E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, *Journal of the ACM* **34** (1987), 596 – 615.

[63] Frieze, A. M., G. Galbiati, F. Maffioli, On the worst-case performance of some algorithms for the asymmetric traveling salesman problem, *Networks* **12** (1982), 23-39.

[64] Gabow, H., An Efficient Implementation of Edmonds' Algorithm for Maximum Matchings in Graph, *J. ACM* **23** (1975), 221-234.

[65] Gabow, H. N., R. E. Tarjan, Faster scaling algorithms for network problems, *SIAM Journal on Computing* **18** (1989), 1013-1036.

[66] Gallo, G., S. Pallotino, Shortest Path Methods: A Unifying Approach, *Math. Program. Study* **26** (1986), 38-64.

[67] Garey, M. R., R. L. Graham and D. S. Johnson, Some NP-complete geometric problems, Proc. 8th ACM Symp. on Theory of Computing, 1976.

[68] Garey, M. R., D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W.H. Freeman & Co, San Francisco, 1979.

[69] Garfinkel, R., Optimal Set Covering: A Survey, in *Perspectives in Optimization: A Collection of Expository Articles* (ed.: Geoffrion), Reading, Mass., Addison- Wesley, 1972.

[70] Garfinkel, R. S. and G. L. Nemhauser, *Integer Programming*, Wiley and Sons, 1972.

[71] Gendreau, M., A. Hertz, G. Laporte, New insertion and postoptimization procedures for the traveling salesman problem, *Operations Research* **40** (1992), 1086-1094.

[72] Gens, G., V., E. V. Levner, Computational complexity of approximation algorithms for combinatorial problems, *Porceedings of MFCS 79,* LNCS **74**, Springer, (1979), 292–300

[73] Gilmore, P. C., Optimal and suboptimal algorithms for the quadratic assignment problem, *SIAM Journal of Applied Mathematics* **10** (1962), 305-313.

[74] Glaab, H., A. Pott, The Hamiltonian p-median problem, *Electronic J. Combin.* **7** (2000), Research Paper 42.

[75] Glover, F., New Results for Reducing Linear Programs to Knapsack Problems, Management Science Report Series, the University of Colorado at Boulder, No. **72-8** (1972).

[76] Golden, B., L. Bodin, T. Doyle, W. Stewart Jr., Approximate traveling salesman algorithms, *Operations Research* **28** (1980), 694-771.

[77] Goldman, A. J., Optimal center location in simple networks, *Transportation Science* **5** (1971), 212-221.

[78] Gomory, R. E., Outline of an algorithm for integer solution to linear programs, *Bull. Amer. Math. Soc.* **64** (1958), 275-278.

[79] Graham R. L., Bounds for certain multiprocessor anomalies, *Bell System Technical Journal* **45** (1966), 1563-1581.

[80] Graham, R. L., E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, *Annals of Discrete Mathematics* **5**, North Holland, Amsterdam, (1979), 287-326.

[81] Greenberg, H., R. Hegerich, A Branch Search Algorithm for the Knapsack Problem, *Management Science* **16** (1970), 327-332.

[82] Gutin, G., A. P. Punnen, *The traveling salesman problem and its variations*, Kluwer Academic Publisher, Dordrecht, 2002.

[83] Hajnal, P., *Gráfelmélet*, POLYGON Kiadó, Szeged, 1997.

[84] Halmos, P. R., H. Vaughon, The marriage problem, *Amer. J. Math.* **72** (1950), 214-215.

[85] Hakimi, S. L., Optimum locations of switching centers and the absolute centers and medians of a graph, *Operations Research* **12** (1964), 450-459.

[86] Handler, G. Y., Minimax location of a facility in an undirected tree graph, *Transportation Science* **12** (1973), 93-96.

[87] Handler, G. Y., Finding two-centers of a tree: The continous case, *Transportation Science* **12** (1978), 93-106.

[88] Handler, G. Y., P. B. Mirchandani, *Location on networks: Theory and algorithms*, MIT Press, Cambridge, 1979.

[89] Hardy, G. G., J. E. Littlewood, and G. Pólya, *Inequalities*, Cambridge University Press, London and New York, 1952.

[90] Horowith, E., S. Sahni, Computing Partitions with Applications to the Knapsack Problem, *Journal of the ACM* **21** (1974), 277-292.

[91] Horowitz, E., S. Sahni: Exact and approximate algorithms for scheduling non identical processors, *Journal of the ACM* **23** (1976), 317–327.

[92] Horowith, E., S. Sahni, *Fundamentals of Computer Algorithms*, Computer Science Press, Potomac, Md., 1978.

[93] Hsu, W., G. L., Nemhauser, Easy and hard bottleneck location problems, *Discrete Applied Mathematics* **1** (1979), 209-215.

[94] Ibarra, O., H., C. E. Kim, Fast approximation algorithms for the knapsack and sum of subset problem, *Journal of the ACM* **22** (1975), 463-468.

[95] Imreh B, Imreh Cs., Imreh Sz: Összefűzési technikák és alkalmazásaik, *Alkalmazott Matematikai Lapok* **22** (2004), 51-62.

[96] Imreh, B., Sz. Imreh, A heuristic method for the asymmetric Hamiltonian p-median problem, *PU.M. A.* **14** (2003), 199–206.

[97] Jackson, J. R., An extension of Johnson's result on job lot scheduling, *Naval Research Logistics Quarterely* **3** (1956), 201-203.

[98] Jeroslow, R. G. and K. O. Kortanek, Dense sets of two variable integer programs requiring arbitrarily many cuts by fractional algorithms, *Man. Sci. Res. Rep.* **174** (1969), Carnegie-Mellon University.

[99] Jewell, W. S., Optimal flow through networks, Interim Technical Report No. 8, Massachusetts Institute of Technology, 1958.

[100] Johnson, D. S., Optimal two- and three-stage production schedules with setup times included, *Naval Research Logistics Quarterely* **1** (1954), 61-68.

[101] Johnson, D. S., L. A. McGeoch, The traveling salesman problem: A case study in local optimization, in: *Local Search in Combinatorial Optimization*, (eds.: E. H. L. , Aarts, J. K. Lenstra), Wiley, New York, 1997, 215-310.

[102] Jordán, T., A. Recski, *Kombinatorikus Optimalizálás*, Műszaki Egyetem és Eötvös Loránd Tudományegyetem, Budapest, 1995.

[103] Jordán, T., A. Recski, D. Szeszlér, *Rendszeroptimalizálás*, Typotex, 2004,

[104] Kameda, T., I. Munro, A $O(|V| \cdot |E|)$ Algorithm for Maximum Matching of Graphs, *Computing* **12** (1974), 91-98.

[105] Kariv, O., S. L. Hakimi, An algorithmic approach to network location problems. Part 2. The p-Median, *SIAM Journal on Applied Mathematics* **37** (1979), 539-560.

[106] Karp, R. M., A patching algorithm for the nonsymmetric traveling salesman problem, *SIAM J. Comput.* **8** (1979), 561-563.

[107] Karp, R. M., Reducibility among Combinatorial Problems, in *Complexity of Computer Computations*, R. E. Miller and T. W. Thatcher, eds., Plenum Press, New York, 1972.

[108] Kellerer, H., U. Pferschy, A new fully polynomial time approximation scheme for the knapsack problem, *Journal on Combinatorial Optimization* **3** (1999), 59-71.

[109] Klee, V., G. J. Minty, How good is the the simplex algorithm?, in: *Inequalities*, III, (ed.: O. Shisha), Academic Press, New York, 1972, 159-175.

[110] Kolesar, P., A Branch and Bound Algorithm for the Knapsack Problem, *Management Science* **13** (1967), 723-735.

[111] Koopmans, T. C., M. J. Beckmann, Assignment problems and the location of economic activities, *Econometria* **25** (1957), 53-76.

[112] Korte, B., J. Vygen, *Combinatorial Optimization, Theory and Algorithms*, – 2. ed.–, Springer Verlag, Berlin,Heidelberg, New York, Barcelona, Hong Kong, London, Milan, Paris, Tokyo, 2002.

[113] Kőnig, D., *Theorie der endlichen und unendlichen Graphen: kombinatorische Topologie der Streckenkomplexe*, Akademische Verlagsgessel schaft, Leipzig, 1936.

[114] Krekó, B., *Optimumszámítás*, Közgazdasági és Jogi Könyvkiadó, Budapest, 1972.

[115] Kruskal, J. B., On the shortest spanning subtree of a graph and the traveling salesman problem, *Proceedings of the American Mathematical Society* **7** (1956), 48-50.

[116] Kuhn, H. W., The Hungarian method for the assignment problems, *Naval. Res. Logist Quart.* **2** (1955), 83-97.

[117] Kuhn, H. W., Variants of the Hungarian method for assignment problems, *Naval. Res. Logist Quart.* **3** (1956), 253-258.

[118] Kuhn, H. W., A magyar módszer eredetéről, *Szigma* **23** (1992), 113-118.

[119] Land, A. H. and A. G. Doig, An automatic method for solving discrete programming problems, *Econometria* **28** (1960), 497-520.

[120] Lawler, E. L., Optimal sequencing of a single machine subject to precedence constraints, *Management Science* **19** 1973, 544-546.

[121] Lawler, E. L., Recent results in theory of machine scheduling, *Mathematical Programming, The State of the Art*, eds.: A. Bachem, B. Korte, M. Grötschel, Springer-Verlag, Berlin-Heidelberg- New York-Tokyo, 1983, 202-234.

[122] Lawler, E. L., *Kombinatorikus optimalizálás*, Műszaki Könyvkiadó, Budapest, 1982.

[123] Lawler, E. L., J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley & Sons, (1985).

[124] Lawler, E. L., The quadratic assignment problem, *Management Science* **9** (1963), 586-599.

[125] Lemke, C. E., H. Salkin, K. Spielberg, Set covering by single branch enumeration with linear programming subproblems, *Operations Research* **19** (1971), 998-1022.

[126] Lin, S., Computer Solutions of the Traveling Salesman Problem, *Bell Syst. Tech. J.* **44** (1965), 2245-2269.

[127] Lin, S., B. W. Keringhan, An Effective Heuristic Algorithm for the Travelling-Salesman Problem, *Operations Research* **21** (1973), 498-516.

[128] Little, J. D. C., K. G. Murty, D. W. Sweeny, C. Karel, An algorithm for the traveling salesman problem, *Operations Research* **11** (1963), 979-989.

[129] Lovász, L., M. D. Plummer, *Matching Theory*, Akadémiai Kiadó, Budapest 1986, and North-Holland, Amsterdam 1986.

[130] Love, R. F., J. G. Morris, G. O. Wesolowsky, *Facilities Location*, North-Holland, New York-Amsterdam-London, 1988.

[131] Makl, K. T., A. J. Morton, A modified Lin-Kernighan traveling-selesman heuristic, *Operations Research Letters* **13** (1993), 127-132.

[132] Martello, S., P. Toth, *Knapsack Problems, Algorithms and Computer Implementations*, John Wiley & Sons Ltd., Chichester - New York - Brisbane - Toronto - Singapure, 1970.

[133] Martello, S., P. Toth, Algorithm 37, Algorithm for the solution of the $0-1$ Single Knapsack Problem, *Computing* **21** (1978), 81-86.

[134] Martello, S., P. Toth, The $0-1$ Knapsack Problem, in: *Combinatorial Optimization* ed. Christofides, et al., Wiley and Sons, 1979.

[135] Martins, E. Q. V., An algorithm for ranking paths that may contain cycles, *European Journal of Operations Resarch* **18** (1984), 123-130.

[136] Marton, L., Optimális úthálózatok meghatározása, *KTMF Tudományos Közlemények* **2** (1977).

[137] Marton, L., Minimálisút algoritmusok közlekedési hálózatokra, *KTMF Tudományos Közlemények* **2** (1979), 219-222.

[138] Marton, L., Egy címkézési eljárás a legrövidebb utak fájának meghatározására ritka hálózatokban *Alkalmazott Matematikai Lapok* **19** (1999), 115-132.

[139] Marton, L., Modelling motorway toll in traffic assignment, *Slovak Journal of Civil Engineering* **4** (1998), 22-26.

[140] Mathews, G., On the partition of Numbers, in: *Proc. of the London Mathematical Society* **28** (1897), 486-490.

[141] Micali, S., V. V. Vazirani, An $O(\sqrt{|V|}\cdot|E|)$ Algorithm for Finding Maximum Matching in General Graphs, *Proc. 21st Annual Symp. on Foundations of Cmputer Science*, IEEE, Long Beach, California (1980), 17-27.

[142] Mirchandani, P. B., R. L. Francis, *Discrete Location Theory*, John Wiley & Sons, Inc., New York-Chichester–Brisbane-Toronto-Singapore, 1990.

[143] Mitchell, J., Guillotine subdivisons approximate polygonal subdivisions: a simple polynomial time approximation scheme for geometric TSP, k-MST and related problems, *SIAM Journal on Computing* **28** (1999), 1298–1309.

[144] Munkres, J., Algorithms for the assignment and transportation problems, *J. SIAM* **5** (1957), $32-38$.

[145] Murty, K. G., An algorithm for ranking all the assignments in order of increasing cost, *Operations Research* **16** (1968), 682-687.

[146] Nauss, R. M., An Efficient Algorithm for the $0-1$ Knapsack Problem, *Management Science* **23** (1976), 27-31.

[147] Nemhauser, G. L., L. A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley & Sons, New York-Chichester-Brisbane-Toronto-Singapore, 1988.

[148] Niven, I., H. S. Zuckerman, *Bevezetés a számelméletbe*, Műszaki Könyvkiadó, 1978.

[149] Papadimitriou, C. H., K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, 1982.

[150] Papadimitriou, C. H., *Számítási bonyolultság*, Novodat Bt, Győr, 1999.

[151] Pukler, A., Vasúti kocsiáramlatok optimalizálása, *SZIKTMF Tudományos Közlemények* **13** (1989), 145-149.

[152] Reinelt, G., *The Traveling Salesman: Computational Solutions for TSP Applications*, Springer-Verlag, Berlin, 1994.

[153] Rinnooy Kan, A. H. G., Report of the session on SCHEDULING, Annals of Discrete Mathematics **5**, North Holland, Amsterdam, (1979), 423-426.

[154] Rónyai, L., G. Iványosi, R. Szabó, *Algoritmusok*, Typotex Kiadó, Budapest, 1998.

[155] Rosenkrantz, D. J., R. E. Stearns, P. M. Lewis, An analysis of several heuristics for the traveling salesman problem, *SIAM J. Comput.* **6** (1977), 563-581.

[156] Rubin, D. S., On the unlimited number of faces in integer hulls of linear programs with a single constraint, *Operations Research* **18** (1970), 940-946.

[157] Sahni, S. and T. Gonzalez, NP-complete approximation problems, *J. Assoc. Comput. Mach.* **23** (1976), 555-565.

[158] Salkin, H., C. DeKluyver, The Knapsack Problem: A Survey, Department of Operations Research Technical Report, Case Western Reserve University, No. **281** (1972).

[159] Salkin, H. M. and K. Mathur, *Foundations of Integer Programming*, North-Holland, 1989.

[160] Schuurman, P., G. J. Woeginger, Approximation Schemes - A Tutorial, megjelenés alatt, elérhető a szerző honlapján.

[161] Sgall, J., On-line scheduling, Chapter 8 (196–231) in Fiat A., G. J. Woeginger (eds.) *Online algorithms: The State of the Art* Vol. 1442 of Lecture Notes in Computer Science, Springer-Verlag, Berlin - Heidelberg, 1998.

[162] Shapiro, D. M., *Algorithms for the Solution of the Optimal Cost and Bottleneck Traveling Salesman Problem*, Sc.D. thesis, Washington University, St. Louis, 1966.

[163] Shier, D. R., Computational Experience with an Algorithm for Finding the k Shortest Path in a Network, *J. Res. Natl. Bur. Stand.* **78B** (1974), 139-165.

[164] Shier, D. R., Iterative Methods for Determining the k Shortest Path in a Network, *Networks* **6** (1976), 205-230.

[165] Smith, T. H. C., V. Srinivasan, G. L. Thompson, Computational performance of three subtour elimination algorithms for solving asymmetric traveling salesman problems, *Ann. Discrete Math.* **I** (1977), 495-506.

[166] Smith, W. E., Various optimizers for single-stage production, *Naval Research Logistics Quarterly* **3** 1956, 59-66.

[167] Spitzer, M., Solution to the crew scheduling problem, presented at the first AGIFORS Symposium, October 1961.

[168] Syslo, M. M., N. Deo, J. S. Kowalik, *Discrete Optimization Algorithm with Pascal Programs*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[169] Taha, A. H., *Integer Programming, Theory, Applications and Computations*, Academic Press, New York, 1975.

[170] Thisse, J. F., H. G. Zoller, eds., *Location Analysis of Public Facilities*, North-Holland, Amsterdam, 1983.

[171] Tucker, A., On directed graphs and integer programs, IBM Mathematical Research Project Technical Report, Princeton University, 1960.

[172] Vazirani, V., *Approximation Algorithms*, Springer-Verlag, Berlin, 2001

[173] Vízvári B., Ütemezéselmélet, *Informatikai Algoritmusok*, szerk.: Iványi A., ELTE Eötvös Kiadó, Budapest, 2004, 364–415.

[174] Warshall, S., A Theorem on Boolean matrices, *J. ACM* **9** (1952), 11-12

[175] Weber, A., Über den Standort der Industrien, Tübingen, 1909, (in German); English Translation: Theory of the location of industries, (C, J. Friedrich, ed. and transl.), Chicago University Press, Chicago, Illinois, 1929.

[176] Weiszfeld, E., Sur un probleme de minimum dans l'espace, *Tohoku Mathematical Journal* **43** (1936), 274-280.

[177] Woeinger, J., When does a dynamic programming formulation guarantee the existence of a fully polynomial time approximation scheme (FPTAS)? *INFORMS Journal on Computing* **12**, (2000), 57–74.

[178] Wilson, R. J., *Introduction to Graph Theory*, 4th edition, Addison-Wesley Pub Co,(ISBN: 0582249937), 1999.

[179] Witzgall C., C. T. Zahn, Jr., Modification of Edmonds' Maximum Matching Algorithm, *J. Res. Nat. Bur. Standards* **69B** (1965), 91-98.

[180] Yen, J. Y., Finding the k shortest loopless paths in a network, *Management Science* **17** (1971), 712-716.

[181] Yudin, D., E. G. Gol'shtein, *Linear Programing Problems of Transportation* (in Russian), NAUKA, 1969.

Tárgymutató