TARTALOMJEGYZÉK

A gazdaságos áramkörök tervezése nagymértékben a
kapcsoló árakörök minimalizálását jelenti.
– H. A. CURTIS *A New Approach to the Design of Switching Circuits* (1854) [15, 161. és 162. oldal]

Egy óriási számítógépnek kell lennie, egyébként kinek sikerülne.
Egyszerűsíteni, egyszerűsíteni.
– HENRY D. THOREAU, *Walden; or, Life in the Woods* (1854) [59, 2. fejezet 17. &]

7.1.2. Boole-függvények kiértékelése

A következő célunk a Boole-függvények hatékony kiértékelésének tárgyalása a 4.6.4. pontbeli polinom kiértékelés stílusában. E téma tanulmányozásának természetes útja az alapművelet-lánc vizsgálata, ahogy azt a fenti pontban a polinom láncok esetén is tettük.

Az $(x_1, \ldots, x_n)$-től függő, n változós Boole-függvényhez rendelt *Boole-lánc* egy olyan $(x_{n+1}, \ldots, x_{n+r})$ sorozat, ahol minden lépés két korábbi lépés kombinációja:

$$x_i \;=\; x_{j(i)} \circ_i x_{k(i)}, \qquad n+1 \le i \le n+r\text{-re}, \tag{1}$$

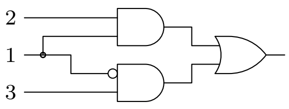
ahol $1 \le j(i) < i$ és $1 \le k(i) < i$, és ahol $\circ_i$ a **??**. táblázatban lévő tizenhat bináris operátor valamelyike. Például, ha $n = 3$ a

$$\begin{aligned} x_4 &= x_1 \wedge x_2 \\ x_5 &= \bar{x}_1 \wedge x_3 \\ x_6 &= x_4 \vee x_5 \end{aligned} \qquad \text{és} \qquad \begin{aligned} x_4 &= x_2 \oplus x_3 \\ x_5 &= x_1 \wedge x_4 \\ x_6 &= x_3 \oplus x_5 \end{aligned} \tag{2}$$

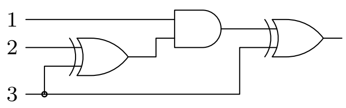
láncok mindegyike a „mux” vagy „elágazás” (if-then-else) függvényt $x_6 = (x_1?\ x_2\colon x_3)$ értékeli ki, ami az x_2 vagy x_3 értéket adja, attól függően, hogy x_1 értéke 1 (igaz) vagy 0 (hamis). (Megjegyezzük, hogy (2) bal oldalán egy NEM-DE (NOT-BUT) művelet leírására az egyszerűsített „$x_5 = \bar{x}_1 \wedge x_3$” jelölést alkalmazzuk a **??**. táblázatbeli „$x_5 = x_1 \overline{\subset} x_3$” helyett. A lényeg annyi, hogy a jelölésektől eltekintve a Boole-lánc minden lépése két korábbi eredmény Boole-kombinációja.)

A Boole-láncok természetes módom megfeleltethetőek elektronikus áramköröknek, hogy a lánc egy lépése egy két bemenetű és egy kimenetű „kaput” jelent.

A villamosmérnökök a (2)-beli Boole-láncokat például az alábbi áramköri diagramokkal ábrázolják:



és

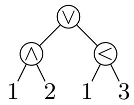
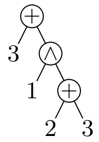


(3)

A mérnököknek olyan gazdaságos áramköröket kell tervezniük, amelyek több technológiai megszorításnak eleget tesznek; például figyelembe kell venni az egyes kapuk árát, a kimeneteket esetleg erősítővel kell ellátni a többszörös felhasználhatóság érdekében, megkövetelhető, hogy az áramkör vezetékezése teljesen vagy majdnem teljesen síkban legyen elhelyezhető, bizonyos huzalhosszak rövidek legyenek stb. Ebben a könyvben azonban a szoftver és nem a hardver érdekel minket, ezért a fent említett dolgokkal nem foglalkozunk.

Számunkra minden kapu ugyanannyiba kerül és minden kimenet tetszőleges sokszor felhasználható. (Röviden szólva a Boole-láncok olyan áramkörökből épülnek fel, amelyekben minden kapunak két bemenete van és korlátozás nélkül terhelhető.)

A továbbiakban a a Boole-láncokat a (3) áramköri diagramok helyett olyan bináris fákkal ábrázoljuk, amilyet például (4)-ben láthatunk:


és

. (4)

Az ilyen bináris fákban lehetnek átfedő részfák, amennyiben a lánc közbenső lépései egynél többször is előfordulnak. Minden belső csúcs cinkéje egy binér műveleti jel lesz; a levelek cinkéje pedig egy k egész szám, ami az x_k változót jelöli. (4) baloldali fájában a ‘$\olessthan$ ’$<$’ címke a NEM-DE műveletet jelöli, ami a $\bar{x} \wedge y = [x < y]$. Hasonlóan jelölheti a ‘$\ogreaterthan$’ címke a DE-NEM (BUT-NOT) operátort, ami $x \wedge \bar{y}$ és a ‘$>$’ csúcscímkével ábrázolható.

Lehetséges, hogy különböző Boole-láncok fa diagramja egyforma. Például (4) baloldali fája reprezentálhatja a következő láncot is:

$$x_4 = \bar{x}_1 \wedge x_3, \qquad x_5 = x_1 \wedge x_2, \qquad x_6 = x_5 \vee x_4 .$$

A fa csúcsainak bármely topológiai elrendezése ekvivalens láncot eredményez.

Adott n-változós f Boole-függvény esetén legtöbbször olyan Boole-láncot keresünk, ahol $x_{n+r} = f(x_1, \ldots, x_n)$ és r a lehető legkisebb. Egy f függvény ***kombinatorikus bonyolultsága*** $C(f)$, az f-et kiszámító legrövidebb lánc hossza. Egyszerűség kedvéért $C(f)$-et „az f költségének” nevezzük. A fenti példabeli mux függvény költsége 3, mivel minden eset kipróbálásával megmutatható, hogy 2 hosszúságú Boole-lánccal nem lehet előállítani.

f-nek a 7.1.1. pontban tárgyalt DNF vagy KNF alakú ábrázolása ritkán ad komolyabb információt $C(f)$-ről, mivel azoknál sokkal hatékonyabb kiszámítási sémák is léteznek. Például a (**??**)-at követő elemzésben láttuk, hogy a többé-kevésbé véletlen, 1100 1100 1001 0000 1111 igazságtáblájú négyváltozós függvénynek nincs rövidebb DNF alakja, mint

$$(\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3) \vee (\bar{x}_1 \wedge \bar{x}_3 \wedge \bar{x}_4) \vee (x_2 \wedge x_3 \wedge x_4) \vee (x_1 \wedge x_2) . \tag{5}$$

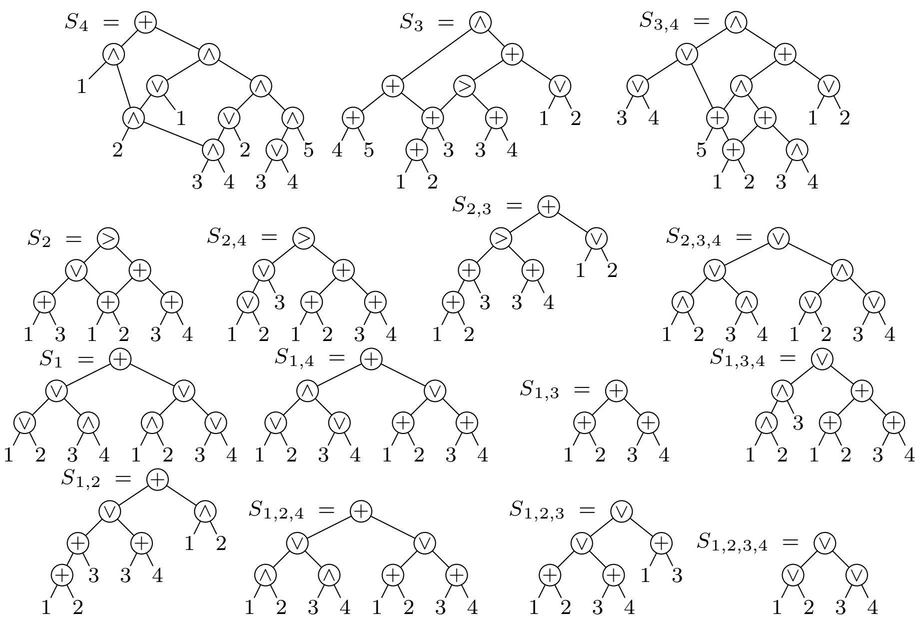
Ez a formula egy 10 lépéses Boole-láncnak felel meg. Ugyanakkor ez a függvény kifejezhető ügyesebben is:

$$\big(((x_2 \wedge \bar{x}_4) \oplus \bar{x}_3) \wedge \bar{x}_1\big) \oplus x_2 , \tag{6}$$

ennek a bonyolultsága legfeljebb 4.

Hogyan lehet a (6)-hoz hasonló, nem nyilvánvaló formulákat megtalálni? Látni fogjuk, hogy a négyváltozós függvényekhez egy számítógép képes megtalálni a legjobb láncokat nagy mennyiségű munka nélkül. Az eredmények valóban meglepőek, főleg azok számára akiknek komoly tapasztalatai vannak a Boole-algebrában. Jellegzetes példákat láthatunk erre a 7.9. ábrán, ahol azokat a négyváltozós függvényeket mutatjuk, amelyek talán a legnagyobb érdeklődésre tartanak számot, nevezetesen azokat, amelyek változóik minden permutációjára szimmetrikusak.

Tekintsük például azt az $S_2(x_1, x_2, x_3, x_4)$ függvényt, amire a 7.9. ábra alapján ezt



7.9. ábra. Optimális Boole-láncok négy változó szimmetrikus függvényére.

kapjuk:

$$\begin{array}{rl} x_1 & 0000\ 0000\ 1111\ 1111 \\ x_2 & 0000\ 1111\ 0000\ 1111 \\ x_3 & 0011\ 0011\ 0011\ 0011 \\ x_4 & 0101\ 0101\ 0101\ 0101 \\ x_5 = x_1 \oplus x_3 & 0011\ 0011\ 1100\ 1100 \\ x_6 = x_1 \oplus x_2 & 0000\ 1111\ 1111\ 0000 \\ x_7 = x_3 \oplus x_4 & 0110\ 0110\ 0110\ 0110 \\ x_8 = x_5 \vee x_6 & 0011\ 1111\ 1111\ 1100 \\ x_9 = x_6 \oplus x_7 & 0110\ 1001\ 1001\ 0110 \\ x_{10} = x_8 \wedge \bar{x}_9 & 0001\ 0110\ 0110\ 1000\,. \end{array} \tag{7}$$

Itt az igazságtáblákat látjuk, amelyekkel könnyen ellenőrizhető a kiszámítás minden lépése. Az x_8 lépés egy olyan függvényt jelent, ami akkor igaz, ha $x_1 \neq x_2$ vagy $x_1 \neq x_3$; és $x_9 = x_1 \oplus x_2 \oplus x_3 \oplus x_4$ az $(x_1+x_2+x_3+x_4) \mod 2$ paritásfüggvény. Ezért a végeredmény, x_{10} pontosan akkor igaz, ha $\{x_1, x_2, x_3, x_4\}$ közül pontosan kettőnek az értéke 1; ezek azok az esetek, amelyek kielégítik x_8-at és a paritásuk páros.

A 7.9. ábra számos kiszámítási sémája igazolható intuitív úton. Néhány lánc azonban, mint például az $S_{1,4}$ függvényhez tartozó, kifejezetten érdekes.

Figyeljük meg, hogy (7)-ben az x_6 részeredményt kétszer használjuk fel. Nincs is olyan hatlépéses lánc az $S_2(x_1, x_2, x_3, x_4)$ függvényre, amelyben valamelyik részkifejezést ne kellene kétszer felhasználni; S_2 valamennyi legrövidebb algebrai kifejezésének, köztük olyan elegáns, szimmetrikus formulának, mint

$$\big((x_1 \wedge x_2) \vee (x_3 \wedge x_4)\big) \oplus \big((x_1 \vee x_2) \wedge (x_3 \vee x_4)\big)\,, \tag{8}$$

7 a költsége. De a 7.9. ábra mutatja, hogy a többi négyváltozós szimmetrikus függvény optimálisan kiértékelhető „tiszta” bináris fákkal, amelyekben csak a változókat ábrázoló levelek fordulnak elő többször.

Egy $f(x_1, \ldots, x_n)$ Boole-függvény $L(f)$ *hosszán* az f-et leíró legrövidebb kifejezésben lévő kétváltozós műveleti jelek száma. Világos, hogy $L(f) \geq C(f)$; és $n \leq 3$ esetén könnyen ellenőrizhető, hogy $L(f) = C(f)$, ha megnézzük a (**??**)-ben a 3 változós függvények tizennégy alaptípusát. Láttuk azonban, hogy $n = 4$-re $L(S_2) = 7$ nagyobb, mint $C(S_2) = 6$, és nagy n esetén $L(f)$ szinte mindig lényegesen nagyobb, mint $C(f)$ (lásd a 49. gyakorlatot).

Egy f Boole-függvény $D(f)$ *mélysége* a bonyolultság másik fontos mértéke. Azt mondjuk, hogy egy Boole-lánc mélysége a lánc fadiagramjában a leghosszabb lefelé haladó út hossza. $D(f)$ pedig az f Boole-láncaiban lévő leghosszabb utak hosszainak minimuma. A 7.9. ábrán látható láncok költsége és mélysége is minimális – kivéve az $S_{2,3}$ és $S_{1,2}$ eseteket, ahol nem lehet elérni egyszerre a 6 költséget és a 3 mélységet. A

$$S_{2,3}(x_1, x_2, x_3, x_4) = \big((x_1 \wedge x_2) \oplus (x_3 \wedge x_4)\big) \vee \big((x_1 \vee x_2) \wedge (x_3 \oplus x_4)\big) \tag{9}$$

formula mutatja, hogy $D(S_{2,3}) = 3$, és hasonló formula áll elő $S_{1,2}$-re is.

Optimális láncok $n = 4$-re. A 4 változós függvények végigszámolhatók, mivel mindössze $2^{16} = 65\,536$ az igazságtáblák száma. Valójában az igazságtábláknak csak a felét kell feldolgozni, mivel egy f függvény $\bar{f}$ negáltjának a költsége, hossza és mélysége ugyanannyi, mint f-nek.

Azt mondjuk, hogy $f(x_1, \ldots, x_n)$ *normalizált*, ha $f(0, \ldots, 0) = 0$ és általában az

$$f(x_1, \ldots, x_n) \,\oplus\, f(0, \ldots, 0) \tag{10}$$

az f „normalizáltja”. Bármely Boole-láncot lehet úgy normalizálni, hogy minden lépését normalizáljuk, és megfelelő módon megváltoztatjuk a műveleti jeleket. Ha ugyanis $(\hat{x}_1, \ldots, \hat{x}_{i-1})$ az $(x_1, \ldots, x_{i-1})$ normalizáltja és ha $x_i = x_{j(i)} \circ_i x_{k(i)}$, mint az (1)-ben, akkor $\hat{x}_i$ az $\hat{x}_{j(i)}$ és $\hat{x}_{k(i)}$ bináris függvénye. (A 7. gyakorlatban mutatunk erre egy példát.) Ezért az általánosság megsértése nélkül elegendő normalizált Boole-láncokkal foglalkozni.

Megjegyezzük, hogy egy Boole-lánc akkor és csak akkor normalizált, ha minden $\circ_i$ kétváltozós operátora normalizált. Csak nyolc normalizált kétváltozós operátor van és ezek közül is három, nevezetesen a $\perp$, $\llcorner$ és R triviálisak. Így feltehetjük, hogy minden fontosabb Boole-lánc öt műveleti jellel ($\wedge$, $\overline{\subset}$, $\overline{\supset}$, $\vee$ és $\oplus$) épül fel. Ezeket a műveleti jeleket a 7.9. ábrában rendre Ⓐ, ⧀, ⧁, Ⓥ, $\oplus$-szal jelöljük. Ezenkívül feltesszük, hogy minden lépésben $j(i) < k(i)$.

A normalizált négyváltozós függvények száma $2^{15} = 32{,}768$ és hosszukat megállapíthatjuk az összes 0, 1, 2 stb. hosszúságú függvény módszeres felsorolásával. Valóban $L(f) = r$ azt jelenti, hogy $f = g \circ h$ valamely g-re és h-ra, ahol $L(g) + L(h) = r - 1$ és $\circ$ az öt nem triviális normalizált művelet valamelyike. Ennek megfelelően az eljárás lehet a következő.

L algoritmus (*normalizált hosszak keresése*). Az algoritmus meghatározza $L(f)$-et bármely normalizált $0 \leq f < 2^{2^n-1}$ igazságtábla esetén, előállítva az összes nemnulla r ($r \geq 0$) hosszúságú, nemnulla normalizált függvény listáját.

L1. (Kezdeti értékek beállítása.) Legyen $L(0) \leftarrow 0$ és $L(f) \leftarrow \infty$ az $1 \leq f < 2^{2^n-1}$ esetben. Legyen az $1 \leq k < n - 1$ értékekre $L(x_k) \leftarrow 0$ és a nulladik listába tegyük be x_k-t, ahol

$$x_k \;=\; (2^{2^n} - 1)/(2^{2^{n-k}} + 1) \tag{11}$$

az x_k igazságtáblája. (Lásd a 8. gyakorlatot.) Végül legyen $c \leftarrow 2^{2^n-1} - n - 1$; c azoknak a helyeknek a száma, ahol $L(f) = \infty$.

L2. (Ciklus r-re.) Az L3 lépés végrehajtása az $r = 1,\ 2,\ \ldots$ értékekre addig, amíg az algoritmus befejeződik azzal, hogy c értéke 0 lesz.

L3. (Ciklus j-re és k-ra.) Az L4 lépés végrehajtása a $j = 0,\ 1,\ \ldots$ értékekre és $k = r-1-j$-re, amíg $j \le k$.

L4. (Ciklus g-re és h-ra.) Az L5 lépés végrehajtása a j-edik lista minden g és a k-adik lista minden h elemére. (Ha $j = k$, akkor elegendő azokat a h függvényeket tekinteni, amelyek *követik* g-t a k-adik listában.)

L5. (Ciklus f-re.) Az L6 lépés végrehajtása $f = g\&h$, $f = \bar{g}\&h$, $f = g\&\bar{h}$, $f = g|h$, és $f = g \oplus h$-ra. (Itt $g\&h$ a g és h egész számok bináris felírásának bitenkénti ÉS műveletének eredményét jelöli; az igazságtáblákat egész számok bináris alakjával ábrázoljuk.)

L6. (Új-e f?) Ha $L(f) = \infty$, akkor legyen $L(f) \leftarrow r$, $c \leftarrow c-1$, és tegyük f-et az r listába. Az algoritmus befejeződik, ha $c = 0$. ■

A 10. gyakorlat mutatja, hogy hasonló eljárással ki lehet számítani az összes $D(f)$ mélységet is.

Az L algoritmust lehet úgy módosítani, hogy egy kis többletmunkával $C(f)$-re jobb felső korlátokat találjon. Ez egy heurisztikus $\phi(f)$ bitvektor kiszámítása, amit f „lábnyomának” nevezünk. Egy normalizált Boole-lánc mindössze $5\binom{n}{2}$ különböző módon kezdődhet, mivel az x_{n+1} első lépés kizárólag az $x_1 \wedge x_2$ vagy $\bar{x}_1 \wedge x_2$ vagy $x_1 \wedge \bar{x}_2$ vagy $x_1 \vee x_2$ vagy $x_1 \oplus x_2$ vagy $x_1 \wedge x_3$ vagy $\ldots$ vagy $x_{n-1} \oplus x_n$ valamelyike lehet. Tegyük fel, hogy $\phi(f)$ egy $5\binom{n}{2}$ hosszúságú bitvektor és $U(f)$ a $C(f)$-nek a következő tulajdonságú felső korlátja: Minden $\phi(f)$-beli 1-bit megfelel valamelyik, az f-et $U(f)$ lépésben kiszámító Boole-lánc első elemének.

Ilyen $(U(f), \phi(f))$ párokat ki lehet számolni az L algoritmus alapstratégiájának kiterjesztésével. Induláskor legyen $U(f) \leftarrow 1$ és $\phi(f)$ legyen minden 1 költségű f-re egy megfelelő $0\ldots010\ldots0$ vektor. Ezután az $r = 2,\ 3,\ \ldots$ értékekre olyan $f = g \circ h$ függvényeket keresünk, ahol $U(g) + U(h) = r - 1$, mint korábban, de két változtatással: (1) Ha g és h lábnyomaiban van legalább egy közös elem, azaz $\phi(g)\&\phi(h) \ne 0$, akkor tudjuk, hogy $C(f) \le r-1$, tehát $U(f)$ csökkenthető, amennyiben az $\ge r$ volt. (2) Ha $g \circ h$ költsége egyenlő az aktuális $U(f)$ felső korláttal (de annál nem kisebb), akkor legyen $\phi(f) \leftarrow \phi(f)|(\phi(g)|\phi(h))$, ha $U(f) = r$, $\phi(f) \leftarrow \phi(f)|(\phi(g)\&\phi(h))$, ha $U(f) = r-1$. A 11. gyakorlat mutatja a kidolgozás részleteit.

Kiderül, hogy ez a heurisztikus lábnyom eléggé hatékony ahhoz, hogy $n = 4$ esetben minden f függvényre megtaláljuk az $U(f) = C(f)$ optimális költségű láncokat. Később látni fogjuk, hogy a lábnyomok sokkal összetettebb kiértékelési problémák esetén is segítségünkre lesznek.

A **??** táblázatnak megfelelően a $2^{16} = 65{,}536$ négyváltozós függvény mindössze 222 különböző osztályhoz tartozik, ha nem veszünk tudomást a változók permutálása és/vagy az értékek komplementálása miatt adódó kisméretű eltérésekről. Az L algoritmus és változatai vezettek a 7.6. táblázatban lévő statisztikai adatokhoz.

***Kiértékelés minimális memóriával.** Tegyük fel hogy az $x_1, \ldots, x_n$ változók értékei n regiszterben jelennek meg és ki akarunk értékelni egy függvényt egy

$$x_{j(i)} \;\leftarrow\; x_{j(i)} \circ_i x_{k(i)}, \qquad (1 \le i \le r) \tag{12}$$

alakú műveletsorozat végrehajtásával, ahol $1 \le j(i) \le n$ és $1 \le k(i) \le n$ és $\circ_i$ egy kétválto-

7.6. táblázat.
ADOTT BONYOLULTSÁGÚ NÉGYVÁLTOZÓS FÜGGVÉNYEK SZÁMA

$C(f)$	Osztályok	Függvények	$L(f)$	Osztályok	Függvények	$D(f)$	Osztályok	Függvények
0	2	10	0	2	10	0	2	10
1	2	60	1	2	60	1	2	60
2	5	456	2	5	456	2	17	1458
3	20	2474	3	20	2474	3	179	56456
4	34	10624	4	34	10624	4	22	7552
5	75	24184	5	75	24184	5	0	0
6	72	25008	6	68	24640	6	0	0
7	12	2720	7	16	3088	7	0	0

zós művelet. A számítás végén a kívánt függvényérték a regiszterek valamelyikében jelenik meg.

Például $n = 3$-ra a

$$\begin{array}{ll} & (x_1 = 00001111 \quad x_2 = 00110011 \quad x_3 = 01010101) \\ x_1 \leftarrow x_1 \oplus x_2 & (x_1 = 00111100 \quad x_2 = 00110011 \quad x_3 = 01010101) \\ x_3 \leftarrow x_3 \wedge x_1 & (x_1 = 00111100 \quad x_2 = 00110011 \quad x_3 = 00010100) \\ x_2 \leftarrow x_2 \wedge \bar{x}_1 & (x_1 = 00111100 \quad x_2 = 00000011 \quad x_3 = 00010100) \\ x_3 \leftarrow x_3 \vee x_2 & (x_1 = 00111100 \quad x_2 = 00000011 \quad x_3 = 00010111) \end{array} \tag{13}$$

négylépéses sorozat kiszámítja $\langle x_1x_2x_3\rangle$ mediánját és az eredményt az x_3 eredeti helyén találjuk. (Itt igazságtábla alakban jelenik meg a nyolc – a műveletek előtt és után lehetséges – regisztertartalom.)

Ha a helyzetet visszafelé haladva elemezzük, akkor a számítást ellenőrizhetjük úgy, hogy mindig csak egy igazságtáblával dolgozunk, így nem kell mind a hármat kezelni. Jelölje $f_l(x_1,\ldots,x_n)$ a sorozat l, $l+1$, ..., r-edik lépéseivel kiszámított függvényt, amiből elhagyjuk az első $l-1$ lépést. Példánkban az x_3-beli eredmény a három lépés, $x_3 \leftarrow x_3 \wedge x_1$, $x_2 \leftarrow x_2 \wedge \bar{x}_1$, $x_3 \leftarrow x_3 \vee x_2$, után kapott $f_2(x_1,x_2,x_3)$ lesz. Így a négy lépésben az x_3 regiszterben kiszámított függvény a következő:

$$f_1(x_1,x_2,x_3) \;=\; f_2(x_1 \oplus x_2, x_2, x_3)\,. \tag{14}$$

Hasonlóan $f_2(x_1,x_2,x_3) = f_3(x_1,x_2,x_3 \wedge x_1)$, $f_3(x_1,x_2,x_3) = f_4(x_1,x_2 \wedge \bar{x}_1,x_3)$, $f_4(x_1,x_2,x_3) = f_5(x_1,x_2,x_3 \vee x_2)$, és $f_5(x_1,x_2,x_3) = x_3$. Ezért az igazságtáblákon megfelelő módon haladva visszafelé f_5-ből f_4-be, ..., f_1-be jutunk.

Például tegyük fel, hogy az $f(x_1,x_2,x_3)$ függvény igazságtáblája

$$t \;=\; a_0a_1a_2a_3a_4a_5a_6a_7\,;$$

akkor a $g(x_1,x_2,x_3) = f(x_1 \oplus x_2, x_2, x_3)$ igazságtáblája

$$u \;=\; a_0a_1a_6a_7a_4a_5a_2a_3\,,$$

úgy áll elő, hogy a_x-et $a_{x'}$-vel helyettesítjük, ahol

$$\text{ha} \qquad x = (x_1x_2x_3)_2, \qquad \text{akkor} \qquad x' = ((x_1 \oplus x_2)x_2x_3)_2\,.$$

Hasonlóan a $h(x_1, x_2, x_3) = f(x_1, x_2, x_3 \wedge x_1)$ igazságtáblája

$$v \;=\; a_0a_0a_2a_2a_4a_5a_6a_7\,.$$

Ekkor t-ből u-t és v-t kiszámíthatjuk bitenkénti műveletekkel:

$$u = t \oplus \big((t \oplus (t \gg 4) \oplus (t \ll 4))\&(00110011)_2\big)\,, \tag{15}$$

$$v = t \oplus \big((t \oplus (t \gg 1))\&(01010000)_2\big)\,. \tag{16}$$

Legyen f minimális memóriával való legrövidebb kiszámításának a hossza $C_m(f)$. A visszafelé haladó kiszámítás elve azt jelenti, hogy ha ismerjük minden olyan f függvény igazságtábláit, amelyre $C_m(f) < r$, akkor meg lehet találni az összes olyan függvény igazságtábláját, amelyre $C_m(f) = r$. Itt is elegendő a normalizált függvényeket tekinteni. Ekkor minden olyan normalizált g-re, amelyre $C_m(g) = r - 1$, elő tudjuk állítani a mind az $5n(n - 1)$ igazságtáblát

$$g(x_1, \ldots, x_{j-1}, x_j \circ x_k, x_{j+1}, \ldots, x_n) \tag{17}$$

függvény mind az $5n(n - 1)$ igazságtábláját, amelyekhez r költséget rendelünk, ha ez előzőleg nem történt volna meg. A 14. gyakorlat mutatja, hogy ezeket az igazságtáblákat egyszerű bitenkénti műveletekkel ki lehet számítani a g igazságtáblájából .

$n = 4$ esetén 13 kivételével mind a 222 alapfüggvény-típusra $C_m(f) = C(f)$, tehát ki lehet minimális memóriával értékelni a költség növelése nélkül. Jelesül, minden szimmetrikus függvény rendelkezik ezzel a tulajdonsággal – habár a 7.9. ábrából ez nem nyilvánvaló. Öt függvényosztály esetén $C(f) = 5$ vagy $C_m(f) = 6$, nyolc osztálynál $C(f) = 6$ de $C_m(f) = 7$. A legérdekesebb példa ez utóbbiak közül valószínűleg a $(x_1 \vee x_2) \oplus (x_3 \vee x_4) \oplus (x_1 \wedge x_2 \wedge x_3 \wedge x_4)$ függvény, aminek a költsége 6 a

$$x_1 \oplus (x_3 \vee x_4) \oplus \big(x_2 \wedge (\bar{x}_1 \vee (x_3 \wedge x_4))\big) \tag{18}$$

formula miatt és nincs 7-nél rövidebb minimális memóriájú lánc hozzá. (Lásd a 15. gyakorlatot.)

***A minimális költség meghatározása.** $C(f)$ pontos értéke megállapítható, ha megfigyeljük, hogy valamennyi $(x_{n+1}, \ldots, x_{n+r})$ optimális Boole-lánca nyilvánvalóan teljesíti a következő három feltétel legalább egyikét:

i) $x_{n+r} = x_j \circ x_k$, ahol x_j és x_k nem használnak fel közös részeredményeket;

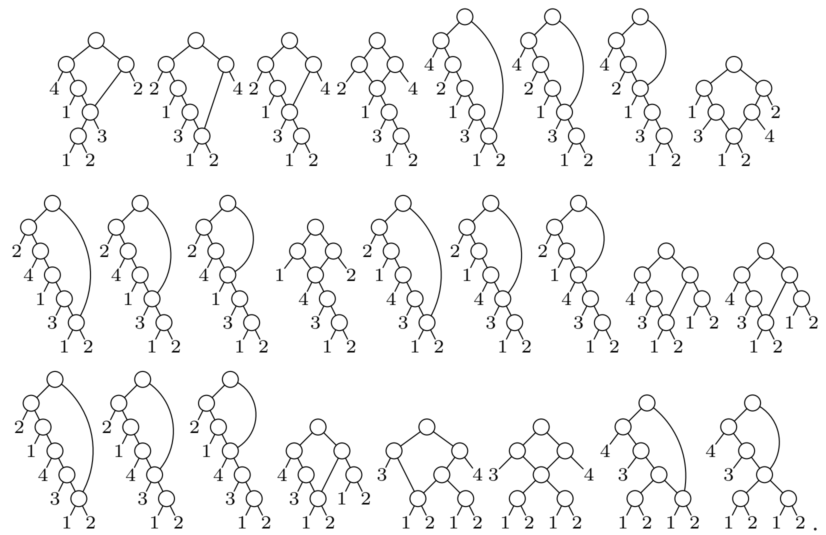
ii) $x_{n+1} = x_j \circ x_k$, ahol x_j és x_k valamelyike nincs felhasználva az $x_{n+2}, \ldots, x_{n+r}$ lépésekben;

iii) A fentiek egyike sem teljesül még akkor sem, ha a lépések sorrendjét megváltoztatjuk.

Az (i) esetben $f = g \circ h$, ahol $C(g) + C(h) = r - 1$. Ezt nevezzük „fentről-lefelé” konstrukciónak. Az (ii) esetben $f(x_1, \ldots, x_n) = g(x_1, \ldots, x_{j-1}, x_j \circ x_k, x_{j+1}, \ldots, x_n)$, ahol $C(g) = r - 1$. Ezt nevezzük „lentről-felfelé” konstrukciónak.

A legjobb olyan láncokhoz, amelyeknél rekurzívan csak fentről-lefelé konstrukciót használnak, az $L(f)$ minimális formulahossz tartozik. A legjobb olyan láncokhoz, amelyeknél rekurzívan csak lentről-felfele konstrukciót használnak, a $C_m(f)$ hosszúságú minimálismemória kiértékelés tartozik. Jobb lehet az eredmény, ha fentről-lefelé és a lentről-felfelé konstrukciót ötvözzük, de nem lehet tudni, hogy $C(f)$-et találtuk-e meg, mivel az (iii) esethez tartozó lánc lehet még ennél is rövidebb.

Szerencsére az ilyen speciális láncok ritkák, mivel nagyon szigorú feltételeknek kell eleget tenniük. Azonban amikor n és r nem túl nagy, ezekről lehet teljes listát készíteni. Például a 19. gyakorlatban megmutatjuk, hogy $r < n + 2$; és ha $n = 4$ és $r = 6$, akkor mindössze a következő 25 olyan lényegesen különböző, speciális lánc létezik, amelyek tovább már nem rövidíthetőek:



Ha végigpróbáljuk minden speciális lánc esetén a fa belső csúcsaihoz a normalizált műveletek 5^r egyenkénti lehetséges hozzárendelését, akkor minden ekvivalenciaosztályból találunk legalább egy f függvényt, amelynek a $C(f)$ minimális költsége csak az (iii) esetben érhető el.

Valóban, ha $n = 4$ és $r = 6$, ez a $25 \cdot 5^6 = 390{,}625$ próbálkozás egyetlen olyan függvényosztályt eredményez, amit nem lehet kiszámolni 6 lépésben egyetlen együttes fentről-lefelé plusz lentről-felfelé lánccal sem. Ez a hiányzó osztály, amit a $(\langle x_1x_2x_3\rangle \vee x_4) \oplus (x_1 \wedge x_2 \wedge x_3)$ részlegesen szimmetrikus függvénnyel jellemzünk, hat lépésben elérhető a fentiekben látható első öt lánc bármelyikéből, megfelelően specializálva ezeket. Például egy lehetőség az

$$\begin{aligned} &x_5 = x_1 \wedge x_2, \quad x_6 = x_1 \vee x_2, \quad x_7 = x_3 \oplus x_5\,, \\ &x_8 = x_4 \wedge \bar{x}_5, \quad x_9 = x_6 \wedge x_7, \quad x_{10} = x_8 \vee x_9\,, \end{aligned} \tag{19}$$

amely az első speciális láncnak felel meg. Mivel minden más függvényre $L(f) \le 7$, az ilyen próbaszámítások minden esetben megadják a minimális költséget.

Történeti megjegyzések. Az első összehangolt kísérleteket az $f(w, x, y, z)$ Boole-függvények optimális kiértékelésére 1951-ben publikálták [1], ahol Howard Aiken csapata ismertetett heurisztikus módszereket, valamint átfogó táblákat az általuk konstruált legjobb logikai áramkörökről. Az ő $V(f)$ költségfüggvényük más volt, mint az általunk tekintett $C(f)$ költség, mivel az a vákuumcsövek vezérlőrácsán alapult. Háromféle kaput használtak: NEM-VAGY-ot, VAGY-ot és ÉS-t és NEM-ÉS-t. Ezek bemenetszáma k és költsége is k volt. Az ilyen kapuk bemenete vagy egy változó vagy egy negált változó vagy egy előző kapu kimenete lehetett. A kiértékelendő függvényt legfelső szinten tetszőleges számú kapu járulékos költség nélküli ÉS-kapcsolatával ábrázolták, ami nem jelentett már költséget.

Ilyen költség-feltételek mellett a függvénynek és a negáltjának nem lehetett ugyanaz a költsége, mivel az ÉS kapuk csak a legfelső szinten fordulhattak elő. Az $x \wedge y$ mint NEM-VAGY$(\bar{x}, \bar{y})$ kiértékelésnek költsége 2, de a $\bar{x} \vee (\bar{y} \wedge \bar{z})$=NEM-ÉS$(x,$VAGY$(y, z))$ kiértékelésnek költsége 4, míg a negáltjáé, ami $x \wedge (y \vee z) = \mathrm{AND}\big(\text{NEM-VAGY}(\bar{x}), \text{VAGY}(y, z)\big)$ csak 3. Ezért a Harvard kutatóinak, 4 változós függvények esetén 402 lényegében különböző függvényosztályt kellett vizsgálni a 222 helyett (lásd a **??**. gyakorlatra adott választ).

Természetesen akkor még kézzel dolgoztak. Azt találták, hogy minden esetben $V(f) < 20$, kivéve a $S_{0,1}(w, x, y, z) \vee \big(S_2(w, x, y) \wedge z\big)$-val ekvivalens 64 függvényt, amit ők 20 vezérlőráccsal értékeltek ki:

$$\begin{aligned} g_1 = \text{NEM-VAGY}(\bar{w}, \bar{x}),\ \ g_2 = \text{NEM-S}(\bar{y}, z),\ \ g_3 = \text{NEM-VAGY}(w, x)\,,& \\ f = \mathrm{S}\big(\text{NEM-S}(g_1, g_2), \text{NEM-S}(g_3, \text{NEM-VAGY}(\bar{y}, \bar{z}))\,,& \\ \text{NEM-VAGY}(\text{NEM-VAGY}(g_3, \bar{y}, z), \text{NEM-VAGY}(g_1, g_2, g_3))\big)\,.& \end{aligned} \tag{20}$$

Hellerman [21] írta az első számítógépes programot bizonyíthatóan optimális áramkörök előállítására, és az adott $f(x, y, z)$ függvény kiértékeléséhez szükséges minimális NOR kapuk számának meghatározására. Megkövetelte, hogy minden kapu minden bemenete vagy egy negálatlan változó vagy egy előző kapu kimenete legyen; a bemenetek és az egy kimenetre kapcsoló bemenetek legfeljebb 3 lehetett. Ha két áramkörben ugyanannyi áramkör szerepelt, akkor azt választotta, ahol az input összeg a legkevesebb. Például kiszámította a $\bar{x} =$ NEM-VAGY(x)-t 1 költséggel; $x \vee y \vee z =$ NEM-VAGY(NEM-VAGY(x, y, z))-t 2 költséggel; $\langle xyz \rangle =$ NEM-VAGY(NEM-VAGY(x, y), NEM-VAGY(x, z), NEM-VAGY(y, z))-t 4 költséggel; $S_1(x, y, z) =$ NEM-VAGY$\big($NEM-VAGY$(x, y, z), \langle xyz \rangle\big)$-t 6 költséggel; stb. Mivel a fan-out korlát nála 3 volt, megmutatta, hogy minden háromváltozós függvény kiértékelhető 7 vagy ennél kevesebb költséggel. Kivétel a $x \oplus y \oplus z = (x \equiv y) \equiv z$ paritás függvény, ahol $x \equiv y$ költsége 4 mivel ez a NEM-VAGY(NEM-VAGY$(x$, NEM-VAGY$(x, y))$, NEM-VAGY$(y$, NEM-VAGY$(x, y)))$.

A villamosmérnökök folytatták a kutatást más költség kritériumok irányába, de 1977-ig a négyváltozós függvények elérhetetlennek tűntek, amikor Liang a 7.6. táblázatban látható $C(f)$ értékeket kiszámolta [30]. Liang publikálatlan munkája az összes olyan lánc tanulmányozásán alapult, amit nem lehetett a fentről-lefelé konstrukcióval redukálni.

***Az $n = 5$ eset.** A **??**. táblázatnak megfelelően az $f(x_1, x_2, x_3, x_4, x_5)$ függvényeknek 616,126 különböző osztálya van. Mivel a számítógépek sebessége elég nagy ahhoz, hogy egy ilyen nagy szám ne legyen elriasztó, a szerző [27] elhatározta, hogy megvizsgálja $C(f)$-et az összes ötváltozós Boole-függvényre. Egy kissé a jó szerencsének is köszönhető, hogy teljes eredményt lehetett elérni, ami a 7.7. táblázat statisztikai adataihoz vezetett.

Ezekhez a számításokhoz az L algoritmust és változatait kellett úgy módosítani, hogy az osztály reprezentánsaival foglalkozzanak a 2^{31} elemű összes normalizált igazságtábla halmaza helyett. A 7.2.1.2–20. gyakorlat módszerével könnyű egy osztály minden függvényét előállítani, sőt ezerszeres gyorsítást érhetünk el, ha annak legalább egy elemét ismerjük. Kiemelnénk a lentről-felfelé módszert, ami lehetővé tette annak megmutatását, hogy $f(x_1 \wedge x_2, x_1 \vee x_2, x_3, x_4, x_5)$ költsége $\leq r$, ha $C(f) = r - 2$. A 10 költségű osztályok megtalálása után a lentről-felfelé és a fentről-lefelé módszerek segítségével sikerült minden esetben megtalálni (hét függvényosztály kivételével) azokat a láncokat, amelyek hossza ≤ 11. A

7.7. táblázat.
ADOTT BONYOLULTSÁGÚ ÖTVÁLTOZÓS FÜGGVÉNYEK SZÁMA

$C(f)$	Oszt.	Függ.	$L(f)$	Oszt.	Függ.	$D(f)$	Oszt.	Függ.
0	2	12	0	2	12	0	2	12
1	2	100	1	2	100	1	2	100
2	5	1140	2	5	1140	2	17	5350
3	20	11570	3	20	11570	3	1789	6702242
4	93	109826	4	93	109826	4	614316	4288259592
5	389	995240	5	366	936440	5	0	0
6	1988	8430800	6	1730	7236880	6	0	0
7	11382	63401728	7	8782	47739088	7	0	0
8	60713	383877392	8	40297	250674320	8	0	0
9	221541	1519125536	9	141422	955812256	9	0	0
10	293455	2123645248	10	273277	1945383936	10	0	0
11	26535	195366784	11	145707	1055912608	11	0	0
12	1	1920	12	4423	31149120	12	0	0

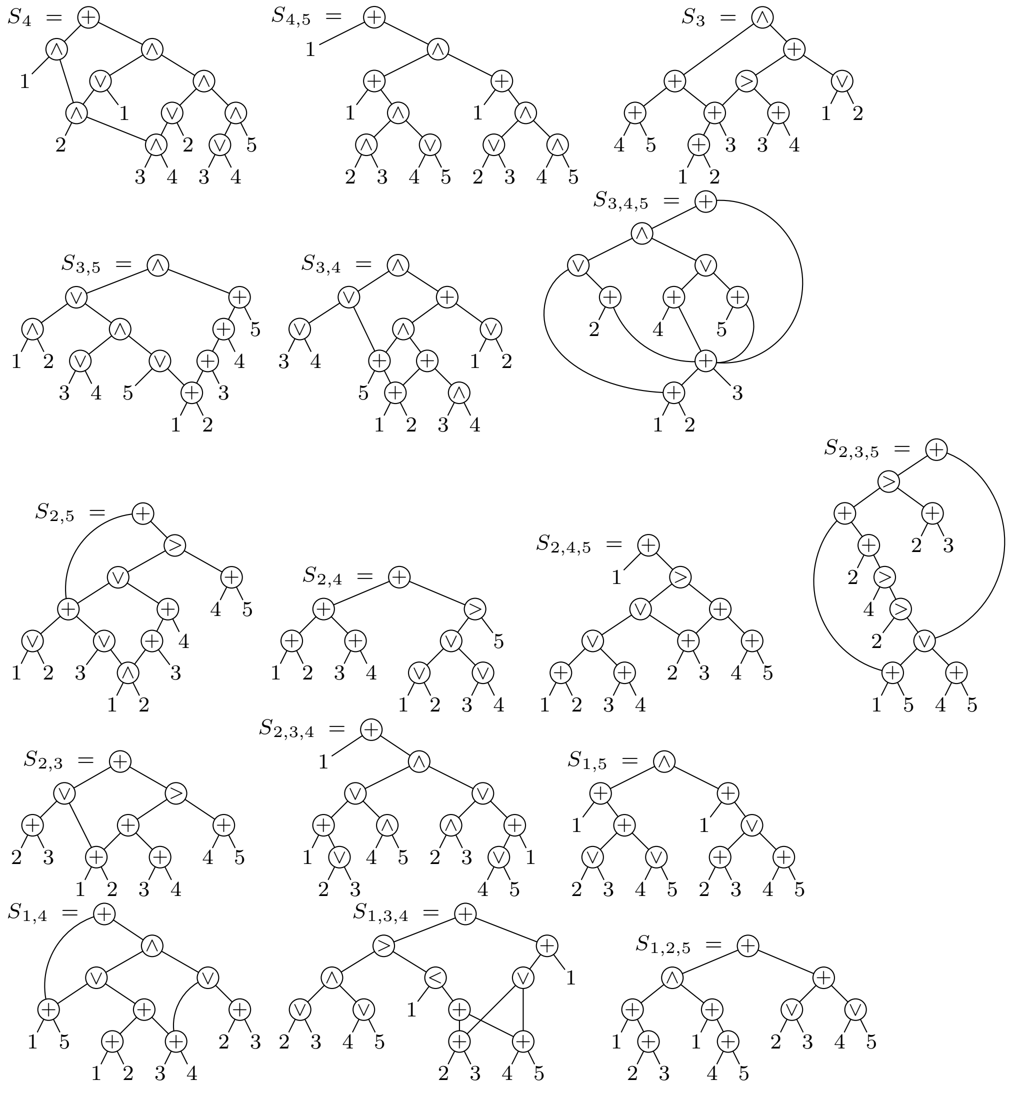
számítás időrabló részében körülbelül 53 millió speciális láncot generáltunk, ahol $n = 5$ és $r = 11$. E láncok mindegyike $5^{11} = 48{,}828{,}125$ függvényt jelent. Remélhetőleg ezek közül egyesek a hét megmaradó különös osztályba tartoznak. Azonban csak hat olyan osztályt találtunk meg, amelyre 11 lépéses megoldások vannak. Az egyedüli hátramaradott, amelynek az igazságtáblája hexadecimális jelölésben `169ae443`, az egyetlen olyan osztály, amelyre $C(f) = 12$ és $L(f) = 12$.

A szimmetrikus függvényekre kapott konstrukciók a 7.10. ábrán láthatók. Ezek közül egyesek meglepően szépek, egyesek csodálatosan egyszerűek és mások egyszerűen meglepőek. (Nézzük például az $S_{2,3}(x_1, x_2, x_3, x_4, x_5)$ 8-lépéses kiszámítását vagy az $S_{2,3,4}$ elegáns formuláját, vagy a nem monoton láncokat $S_{4,5}$ és $S_{3,4,5}$-re.) A 7.7. táblázat azt is mutatja, hogy minden 5 változós függvény mélysége ≤ 4, de a 7.10. ábrán nincs kísérlet sem a mélység minimalizálására. Előfordul, hogy ezeket a szimmetrikus függvényeket ki lehet értékelni minimális memóriával költségnövelés nélkül. De erre a tényre nem ismert egyszerű bizonyítás.

***Többszörös kimenetek.** Gyakran akarunk több különböző $f_1(x_1, \ldots, x_n)$, …, $f_m(x_1, \ldots, x_n)$ Boole-függvényt egyszerre kiértékelni az x_1, …, x_n-re megadott, közös bemeneti értékek mellett. Más szóval gyakran akarunk kiértékelni valamely $y = f(x)$ több bites függvényt, ahol $y = f_1 \ldots f_m$ egy m hosszúságú bináris vektor. Kis szerencsével az egyik $f_j(x_1, \ldots, x_n)$ komponens kiszámításához szükséges munka elvégezhető azokkal a műveletekkel, amelyek egy másik $f_k(x_1, \ldots, x_n)$ komponens kiértékeléséhez szükségesek.

Legyen $C(f) = C(f_1 \ldots f_m)$ annak a legrövidebb Boole-láncnak a hossza, ami kiszámítja az összes nemtriviális f_j függvényt. Pontosabban, az $(x_{n+1}, \ldots, x_{n+r})$ láncnak teljesítenie kell $1 \leq j \leq m$-re a következőt; vagy $f_j(x_1, \ldots, x_n) = x_{l(j)}$ vagy $f_j(x_1, \ldots, x_n) = \bar{x}_{l(j)}$, valamilyen $l(j)$ és $0 \leq l(j) \leq n + r$-re, ahol $x_0 = 0$. Nyilván $C(f) \leq C(f_1) + \cdots + C(f_m)$, de ennél több is igaz.

Például tegyük fel, hogy ki akarjuk kiszámítani z_1, z_0-nak a (21)-ben definiált függvé-



7.10. ábra. Minimális költségű Boole-láncok öt változó szimmetrikus függvényére.

nyét.

$$(z_1 z_0)_2 \;=\; x_1 + x_2 + x_3\,, \tag{21}$$

Ez három Boole-változó összege binárisan ábrázolva. Ekkor

$$z_1 \;=\; \langle x_1 x_2 x_3\rangle \qquad \text{és} \qquad z_0 \;=\; x_1 \oplus x_2 \oplus x_3, \tag{22}$$

$C(z_1) = 4$ and $C(z_0) = 2$ az egyes költségek. Világos, hogy a $C(z_1 z_0)$ együttes költség legfeljebb 5, mivel $x_1 \oplus x_2$ megfelelő első lépés bármelyik z_j bit kiértékeléséhez:

$$\begin{aligned} &x_4 = x_1 \oplus x_2, \quad z_0 = x_5 = x_3 \oplus x_4\,; \\ &x_6 = x_3 \wedge x_4, \quad x_7 = x_1 \wedge x_2, \quad z_1 = x_8 = x_6 \vee x_7\,. \end{aligned} \tag{23}$$

Az összes lehetőség kipróbálása mutatja, hogy $C(z_1z_0) > 4$ és így $C(z_1z_0) = 5$. A (21) áramkört a villamosmérnökök *teljes összeadónak* nevezik, mivel n ilyen blokkot kapcsolnak össze két n-bites szám összeadásához. A (22) speciális esete, amikor $x_3 = 0$, szintén fontos. Ebből egyszerűen

$$z_1 \;=\; x_1 \wedge x_2 \qquad \text{és} \qquad z_0 \;=\; x_1 \oplus x_2 \tag{24}$$

lesz. Ennek 2 a bonyolultsága. A mérnökök „fél összeadónak” hívják annak ellenére, hogy egy teljes összeadó költsége több, mint két fél összeadó költségének összege.

A

$$\begin{array}{r} (x_{n-1}\ldots x_1x_0)_2 \\ (y_{n-1}\ldots y_1y_0)_2 \\ \hline (z_n z_{n-1}\ldots z_1 z_0)_2 \end{array} \tag{25}$$

bináris összeadás általános feladata, hogy a $2n$ darab bemenő $(x_{n-1}\ldots x_1x_0y_{n-1}\ldots y_1y_0)$ Boole-értékből kiszámítsa az $n{+}1$ $(z_n\ldots z_1z_0)$ kimenő Boole-értéket. Erre megoldást adnak a

$$c_{j+1} \;=\; \langle x_jy_jc_j\rangle, \qquad z_j \;=\; x_j \oplus y_j \oplus c_j, \qquad 0 \le j < n\text{-re}, \tag{26}$$

formulák, ahol a c_j-k az „átvitel bitek” és $c_0 = 0$, $z_n = c_n$. Ezért c_1 és z_0 kiszámítására fél összeadót használhatunk , ami után $n{-}1$ teljes összeadó jön a többi c-k és z-k kiszámítására. Ezek teljes költsége $5n{-}3$. Redkin [45] bizonyította, hogy $5n{-}3$ lépés ténylegesen szükséges. Ez egy 35 oldalas indukciós bizonyítás, ami a 2.2.2.3.1.2.3.2.4.3(!) esettel végződik. Ennek az áramkörnek a mélysége $2n-1$, ami gyakorlati párhuzamos számításhoz túl nagy. Ezért komoly erőfeszítéseket tesznek olyan összeadó kieszelésére, amelynek $O(\lg n)$ a mélysége és a költsége is elfogadható. (Lásd a 41–44. gyakorlatokat.)

Most terjesszük ki (21)-et és próbáljunk egy

$$(z_{\lfloor \lg n\rfloor}\ldots z_1z_0)_2 \;=\; x_1 + x_2 + \cdots + x_n \tag{27}$$

általános „oldalösszeget” kiszámítani. Ha $n = 2k{+}1$, akkor felhasználhatunk k teljes összeadót, hogy az összeget leredukáljuk $(x_1 + \cdots + x_n) \bmod 2$-re plusz k bit kétszeresére bitre, mivel minden teljes összeadó 2-vel csökkenti az 1-súlyok számát. Például, ha $n = 9$ és $k = 4$, akkor a számítás:

$$\begin{array}{llll} x_{10} = x_1 \oplus x_2 \oplus x_3, & x_{11} = x_4 \oplus x_5 \oplus x_6, & x_{12} = x_7 \oplus x_8 \oplus x_9, & x_{13} = x_{10} \oplus x_{11} \oplus x_{12}\,, \\ y_1 = \langle x_1x_2x_3\rangle, & y_2 = \langle x_4x_5x_6\rangle, & y_3 = \langle x_7x_8x_9\rangle, & y_4 = \langle x_{10}x_{11}x_{12}\rangle \end{array}$$

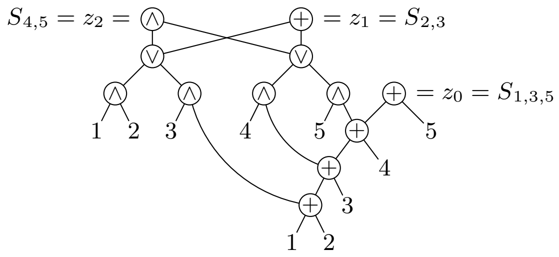
Ekkor azt kapjuk, hogy $x_1 + \cdots + x_9 = x_{13} + 2(y_1 + y_2 + y_3 + y_4)$. Ha $n = 2k$ páros, akkor hasonló egyszerűsítést alkalmazunk, de a végére fél összeadó kerül. A 2 súlyú biteket ugyanígy lehet összegezni. Igy a következő rekurziót kapjuk

$$s(n) \;=\; 5\lfloor n/2\rfloor - 3[n \text{ even}] + s(\lfloor n/2\rfloor), \qquad s(0) = 0, \tag{28}$$

a teljes $z_{\lfloor \lg n\rfloor}\ldots z_1z_0$ kiszámításához szükséges kapuk számára. (A 30. gyakorlatban $s(n)$-re egy zárt formulát találunk.) Azt kapjuk, hogy $s(n) < 5n$ és az

$$\begin{array}{rcccccccccccccccccccc} n = & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 \\ s(n) = & 0 & 2 & 5 & 9 & 12 & 17 & 20 & 26 & 29 & 34 & 37 & 44 & 47 & 52 & 55 & 63 & 66 & 71 & 74 & 81 \end{array}$$

kezdeti értékek azt mutatják, hogy a módszer kis n-ekre egészen hatékony. Például, ha $n = 5$, az eredmény



, (29)

ami a $z_2 = S_{4,5}(x_1, \ldots, x_5)$, $z_1 = S_{2,3}(x_1, \ldots, x_5)$, $z_0 = S_{1,3,5}(x_1, \ldots, x_5)$ három különböző szimmetrikus függvényt pontosan 12 lépésben számítja ki. A 7.10. ábrának megfelelően az $S_{4,5}$ 10-lépéses kiszámítása optimális. Ugyanígy az $S_{4,5}$ 4-lépéses kiszámítása is. Ezenkívül, bár $C(S_{2,3}) = 8$, az $S_{2,3}$-at itt olyan ügyes 10-lépéses módon adjuk meg, ami minden kaput megosztva használ, $S_{4,5}$-ből is egyet.

Megjegyezzük, hogy ezután *minden* szimmetrikus függvény hatékonyan kiszámítható, mivel minden $\{x_1, \ldots, x_n\}$ változós szimmetrikus függvény egy $z_{\lfloor \lg n \rfloor} \ldots z_1 z_0$ változós Boole-függvény. Tudjuk, hogy minden négyváltozós Boole-függvény összetettsége ≤ 7; emiatt bármelyik $S_{k_1,\ldots,k_t}(x_1, \ldots, x_{15})$ szimmetrikus függvény költsége legfeljebb $s(15) + 7 = 62$. Ami meglepő: kis változószám esetén az n-változós szimmetrikus függvények kiértékelése nagyon nehéz, de $n \geq 10$-re ugyanez a feladat a legkönnyebbek közé tartozik.

Szimmetrikus függvények *halmazait* szintén ki lehet hatékonyan számolni. Ha mondjuk egyetlen Boole-lánccal akarjuk kiértékelni az $n+1$ darab $S_k(x_1, \ldots, x_n)$ szimmetrikus függvényt $0 \leq k \leq n$-re, akkor egyszerűen a $z_0, z_1, \ldots, z_{\lfloor \lg n \rfloor}$ változók első $n+1$ *mintermjét* kell kiértékelni. Például, ha $n = 5$, akkor az összes S_k függvényt megadó mintermek a következők: $S_0 = \bar{z}_0 \wedge \bar{z}_1 \wedge \bar{z}_2$, $S_1 = \bar{z}_0 \wedge \bar{z}_1 \wedge z_2$, …, $S_5 = z_0 \wedge \bar{z}_1 \wedge z_2$.

Milyen nehéz n változóhoz a 2^n minterm kiszámítása? Az elektronikai mérnökök ezt a függvényt n-ből-2^n-be *bináris dekódolónak* hívják, mivel az n darab $x_1 \ldots x_n$ bitet egy olyan 2^n elemű $d_0 d_1 \ldots d_{2^n-1}$ sorozatba konvertálja, amelyek közül pontosan egynek lehet 1 az értéke. Az „oszd meg és uralkodj" elvnek megfelelően először az első $\lceil n/2 \rceil$ változón értékeljük ki az összes mintermet, ugyanígy az összes mintermet az utolsó $\lfloor n/2 \rfloor$-n; ezután 2^n ÉS kapuval fejeződik be a munka. A módszer költsége $t(n)$, ahol

$$t(0) = t(1) = 0; \quad t(n) = 2^n + t(\lceil n/2 \rceil) + t(\lfloor n/2 \rfloor) \quad n \geq 2\text{-re}. \tag{30}$$

Igy $t(n) = 2^n + O(2^{n/2})$; ez durván egy kapu mintermenként (lásd a 32. gyakorlatot).

Többkimenetű függvények gyakran segítenek nagyméretű egykimenetű függvények megépítésében. Például, láttuk, hogy a (27)-beli tömörítő összeg, szimmetrikus függvények kiszámítását teszi lehetővé, valamint, hogy a n-ből-2^n-be dekódolónak szintén sok alkalmazása van, attól eltekintve, hogy nagy n esetén 2^n már rendkívül nagy. Hasonló eset az $M_m(x_1, \ldots, x_m; y_0, y_1, \ldots, y_{2^m-1})$ 2^m*-es multiplexer*, ami mint m-bites *memória elérési* *függvény* is ismert; bemenetszám $n = m + 2^m$ és $(x_1 \ldots x_m)_2 = k$ estben az y_k értéket adja. Definíció szerint

$$M_m(x_1, \ldots, x_m; y_0, y_1, \ldots, y_{2^m-1}) \;=\; \bigvee_{k=0}^{2^m-1} (d_k \wedge y_k)\,, \tag{31}$$

ahol d_k az m-ből-2^m bináris dekódoló k-adik kimenete; így a (30)-nak megfelelően M_m-et $2^m + (2^m-1) + t(m) = 3n + O(\sqrt{n})$ kapuval ki lehet értékelni. A 39. gyakorlatban megmutatjuk, hogy a költséget mindössze $2n + O(\sqrt{n})$-re csökkenthetjük. (Lásd a 79. gyakorlatot is.)

***Aszimptotikus tények.** Ha a változók száma kicsi, a teljes kipróbálás módszerével sok olyan eset adódik, ahol a Boole-függvényeket meglepő hatékonysággal ki lehet értékelni. Ezért természetes azt remélni, hogy több változó jelenléte még több ügyes kiértékelési lehetőséget biztosít. Az igazság ennek az ellenkezője, legalábbis a statisztika részéről.

S tétel [49]. *Majdnem minden $f(x_1, \ldots, x_n)$ Boole-függvény költsége meghaladja a $2^n/n$-t. Pontosabban szólva, ha a $c(n, r)$ Boole-függvények összetettsége $\le r$, akkor fennáll a következő:*

$$(r-1)!\,c(n,r) \;\le\; 2^{2r+1}(n+r-1)^{2r}. \tag{32}$$

Bizonyítás. Ha egy függvényt ki lehet számítani $r-1$ lépésben, akkor kiszámítható r-lépéses lánccal is. (Ez az állítás nyilvánvaló, ha $r = 1$; egyébként pedig $x_{n+r} = x_{n+r-1} \wedge x_{n+r-1}$ lehet.) Megmutatjuk, hogy nincs túl sok r-lépéses lánc, emiatt nem túl sok olyan különböző függvényt lehet kiszámítani, ahol a költség $\le r$.

Legyen π az $\{1, \ldots, n+r\}$ egy olyan permutációja, ahol az $1 \mapsto 1$, …, $n \mapsto n$, and $n + r \mapsto n + r$ hozzárendelés rögzített; ilyen permutáció $(r-1)!$ van. Tegyük fel, hogy $(x_{n+1}, \ldots, x_{n+r})$ egy olyan Boole-lánc, amelyben az x_{n+1}, …, x_{n+r-1} közbenső lépések mindegyike legalább egy rész-lépéssorozatban előfordul. Ekkor az

$$x_i \;=\; x_{j'(i)} \circ'_i x_{k'(i)} \;=\; x_{j(i\pi)\pi^-} \circ_{i\pi} x_{k(i\pi)\pi^-}\,, \qquad n < i \le n + r\text{-re}\,, \tag{33}$$

szabállyal definiált permutált láncok különböző π-kre különbözőek lesznek. (Ha π szerint $a \mapsto b$, akkor azt írjuk, hogy $b = a\pi$ és $a = b\pi^-$.) Például, ha π szerint $5 \mapsto 6 \mapsto 7 \mapsto 8 \mapsto 9 \mapsto 5$, akkor a (7) láncból a következő lesz:

$$\begin{array}{cc} \text{Eredeti} & \text{Permutált} \\ x_5 = x_1 \oplus x_3, & x_5 = x_1 \oplus x_2\,, \\ x_6 = x_1 \oplus x_2, & x_6 = x_3 \oplus x_4\,, \\ x_7 = x_3 \oplus x_4, & x_7 = x_9 \vee x_5\,, \\ x_8 = x_5 \vee x_6, & x_8 = x_5 \oplus x_6\,, \\ x_9 = x_6 \oplus x_7, & x_9 = x_1 \oplus x_3\,, \\ x_{10} = x_8 \wedge \bar{x}_9; & x_{10} = x_7 \wedge \bar{x}_8\,. \end{array} \tag{34}$$

Megjegyezzük, hogy a szokásos szabályokkal ellentétben $j'(i) \ge k'(i)$ vagy $j'(i) > i$ vagy $k'(i) > i$ lehetséges. Mindazonáltal a permutált lánc ugyanazt az x_{n+r} függvényt számítja ki, mint korábban és nincs benne olyan ciklus, ami egy bemenetet indirekt módon definiálna önmaga termjeként, mivel x_i permutáltja nem más, mint $x_{i\pi}$.

Mint azt korábban is megjegyeztük most a *normalizált* Boole-láncokkal foglalkozunk. Így a $c(n, r)/2$ normalizált Boole-függvény, aminek a költsége $\le r$, olyan $(r-1)!\,c(n, r)/2$

különböző permutált láncot eredményez, amelyben az $\circ_i$ művelet a $\wedge$, $\vee$, $\overline{\supset}$, or $\oplus$ műveletek valamelyike. Mivel $\circ_i$ kiválasztására négy lehetőség van, valamint $n < i \le n + r$ esetén, $n{+}r{-}1$-féle választás $j(i)$-re és $k(i)$-re, a lehetséges láncok száma legfeljebb $4^r(n{+}r{-}1)^{2r}$. A (32) képletből következik és a tétel függőben lévő állítását $r = \lfloor 2^n/n \rfloor$-re kapjuk meg. (Lásd a 46. gyakorlatot.)

Másrészt a kétkedők számára jó hír, hogy bármely n -változós Boole-függvényt egy kicsivel több, mint $2^n/n$ lépésben ki lehet értékelni, még akkor is, ha a $\oplus$ és a $\equiv$ műveleteket elhagyjuk, amennyiben azt a technikát használjuk, amit Shannon [49] ötlete alapján Lupanov tökéletesített [31] .

A Shannon–Lupanov-közelítés hasznos eredményekhez vezetett, még kis n-re is. Tanulmányozzuk ezt egy kis példán keresztül. Tekintsük a

$$f(x_1,x_2,x_3,x_4,x_5,x_6) \;=\; [(x_1x_2x_3x_4x_5x_6)_2\text{prímszám}]\,, \tag{35}$$

függvényt, ami felismeri az összes 6 bites prímszámot . A függvény igazságtáblája $2^6 = 64$ bitet tartalmaz, amit hagyományosan 4×16-os mátrix alakban adunk meg, majd ezeket a biteket tekintjük az egydimenziós sorozat helyett.

$$\begin{array}{r@{\;}l}
x_3 = & 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\\
x_4 = & 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\\
x_5 = & 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\\
x_6 = & 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\\
x_1x_2 = 00 & \boxed{\begin{array}{l}0\,0\,1\,1\,0\,1\,0\,1\,0\,0\,0\,1\,0\,1\,0\,0\\ 0\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1\\ 0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,1\,0\,0\,0\,1\\ 0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\end{array}} \begin{array}{l}\left.\begin{array}{l}\\ \\ \end{array}\right\}\text{ 1. csoport}\\ \left.\begin{array}{l}\\ \\ \end{array}\right\}\text{ 2. csoport}\end{array}\\
x_1x_2 = 01 & \\
x_1x_2 = 10 & \\
x_1x_2 = 11 &
\end{array} \tag{36}$$

A sorokat két kettes csoportba osztjuk, a sorok mindegyik csoportjának 16 oszlopa van. amelyek négyféle alaptípushoz tartoznak nevezetesen ${}^0_0$, ${}^0_1$, ${}^1_0$, vagy ${}^1_1$. Így a függvény a következőkép fejezhető ki:

$$\begin{aligned}
f(x_1,\ldots,x_6) \;=\;\; & \big([x_1x_2 \in \{00\}] && \wedge\; [x_3x_4x_5x_6 \in \{0010, 0101, 1011\}]\big)\\
\vee\;\; & \big([x_1x_2 \in \{01\}] && \wedge\; [x_3x_4x_5x_6 \in \{0001, 1111\}]\big)\\
\vee\;\; & \big([x_1x_2 \in \{00, 01\}] && \wedge\; [x_3x_4x_5x_6 \in \{0011, 0111, 1101\}]\big)\\
\vee\;\; & \big([x_1x_2 \in \{10\}] && \wedge\; [x_3x_4x_5x_6 \in \{1001, 1111\}]\big)\\
\vee\;\; & \big([x_1x_2 \in \{11\}] && \wedge\; [x_3x_4x_5x_6 \in \{1101\}]\big)\\
\vee\;\; & \big([x_1x_2 \in \{10, 11\}] && \wedge\; [x_3x_4x_5x_6 \in \{0101, 1011\}]\big).
\end{aligned} \tag{37}$$

(Az első sor a ${}^1_0$ típusú 1. csoportnak felel meg, azután jön a ${}^0_1$ típusú 1. csoport stb.; az utolsó sor a ${}^1_1$ típusú 2. csoportnak felel meg.) A $[x_3x_4x_5x_6 \in \{0010, 0101, 1011\}]$-hez hasonló függvény az $\{x_3, x_4, x_5, x_6\}$-ből alkotott három minterm VAGY-ja.

Általában az igazságtáblát megadhatjuk $2^k \times 2^{n-k}$ mátrix alakban, ahol a sorokat l csoportra bontjuk, így minden sorcsoport $\lfloor 2^k/l \rfloor$ sorból áll. Egy m méretű csoportnak 2^m-féle alaptípusa lesz. Képezzük a $(g_{it}(x_1,\ldots,x_k) \wedge h_{it}(x_{k+1},\ldots,x_n))$ konjunkciót minden i-csoportra és minden nemzéró típusú t-re, ahol g_{it} az $\{x_1,\ldots,x_k\}$ összes olyan mintermjének az VAGY-ja, ahol a csoport soraiban t-ben van 1-es, a h_{it} az $\{x_{k+1},\ldots,x_n\}$ összes olyan mintermjének az VAGY-ja amelyek az i csoportba tartozó t típusú oszlopokhoz tartoznak.

Az összes $(g_{it} \wedge h_{it})$ VAGY-ja megadja az $f(x_1, \ldots, x_n)$-et.

Ha megválasztottuk a k és az l paramétereket, ahol $1 \le k \le n-2$ és $1 \le l \le 2^k$, akkor a számítás az $\{x_1, \ldots, x_k\}$ összes mintermjének és $\{x_{k+1}, \ldots, x_n\}$ összes mintermjének kiszámításával kezdődik. Ez $t(k) + t(n-k)$ lépés (lásd (30)). Ezután $1 \le i \le l$-re tekintjük az i-edik csoportot, ami az $(x_1, \ldots, x_k)$-nak az $(i-1)2^k/l \le (x_1 \ldots x_k)_2 < i2^k/l$ értékeiből áll; ez összesen $m_i = \lceil i2^k/l \rceil - \lceil (i-1)2^k/l \rceil$ sor. Előállítjuk az összes g_{it}-t a $t \in S_i$-re, ahol S_i e sorok nemüres részhalmazainak $2^{m_i} - 1$ elemű családja. Az előzőleg kiszámított mintermek diszjunkciójával; vagyis $2^{m_i} - m_i - 1$ OR felhasználásával fejeződik be a feladat. Ezután előállítjuk az összes h_{it}-t a t-típusú oszlopokhoz. Ehhez legfeljebb 2^{n-k} VAGY művelet szükséges minden i csoportban, mivel egy adott t-típusú h függvénybe a mintermek diszjunkciója kerül. Végül kiszámítjuk az $f = \bigvee_{i=1}^{l} \bigvee_{t \in S_i} (g_{it} \wedge h_{it})$-et. Az ÉS műveleteket egyetlen VAGY művelettel számoljuk be minden h_{it}-be. Így a tejes költség legfeljebb

$$t(k) + t(n-k) + (l-1) + \sum_{i=1}^{l} \bigl((2^{m_i} - m_i - 1) + 2^{n-k} + (2^{m_i} - 2)\bigr)\,; \tag{38}$$

Szeretnénk ezt a felső határt minimalizálni k és l alkalmas megválasztásával. Kis n esetére a 52. gyakorlat foglalkozik a legjobb választással. Nagy n-re egy jó választás valószínűleg egy közel optimális láncot eredményez a legtöbb függvényre.

L tétel. Jelölje $C(n)$ a legdrágább n változós Boole-függvény költségét. Ekkor $n \to \infty$ esetén

$$C(n) \;\ge\; \frac{2^n}{n}\Bigl(1 + \frac{\lg n}{n} + O\Bigl(\frac{1}{n}\Bigr)\Bigr)\,; \tag{39}$$

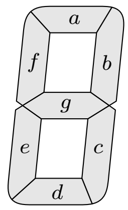
$$C(n) \;\le\; \frac{2^n}{n}\Bigl(1 + 3\frac{\lg n}{n} + O\Bigl(\frac{1}{n}\Bigr)\Bigr)\,. \tag{40}$$

Bizonyítás. A 46. gyakorlat mutatja, hogy a (39) alsó határ az S tétel következménye. Ami a felső határt illeti, Lupanov módszere [33] szerint beállítjuk, hogy $k = \lfloor 2 \lg n \rfloor$ és $l = \lceil 2^k/(n - 3\lg n) \rceil$ teljesüljön. Lásd az 53. gyakorlatot. ■

Jó lánc szintetizálása. Bár a (37) formula nem adja meg a legjobb módot a 6 bites prímszám-felismerő implementálásához, azonban egy eléggé jó stratégiát sugall. Például nem okvetlenül az x_1-et és x_2-t kell a sorokhoz rendelni. Az 51. gyakorlat mutatja, hogy sokkal jobb láncot eredményez, ha a sorokhoz x_5x_6-ot, az oszlopokhoz pedig $x_1x_2x_3x_4$-et rendeljük. Általában sok lehetőség van arra, hogy egy igazságtáblát k változó és $n-k$ változó szerint particionáljunk.

Ezenkívül a 4 változós függvényekkel kapcsolatos tudásunkat felhasználva javíthatunk (37)-en. Ugyanis ahhoz, hogy az $[x_3x_4x_5x_6 \in \{0010, 0101, 1011\}]$-hoz hasonló függvényeket kiértékeljük, nincs arra szükség, hogy először kiszámítsuk $\{x_3, x_4, x_5, x_6\}$ mintermjeit, amennyiben ismerjük a legjobb módot, vázlatosan megadott hasonló függvények kiértékelésére. Másrészt szükség lesz néhány 4 változós függvény szimultán kiértékelésére, ezért a minterm ötlet egyáltalán nem olyan rossz. Lehet-e ezen javítani?

Próbáljunk meg 4 változós függvények adott halmazát kiszámító Boole-lánc szintézisére jó módszert keresni. Az $x_3x_4x_5x_6$-nak az (37)-beli hat függvénye eléggé egyszerű, ezért többet lehet tanulni a napi életből választott érdekesebb példákból.



A *hét szegmenses ábrázolás* ma a legelterjedtebb módja az $(x_1x_2x_3x_4)_2$ négybites számok ábrázolásának a hét jól elhelyezett – látható vagy nem látható állapotú – szegmens segítségével. A szegmensek (a, b, c, d, e, f, g) tradicionális megnevezését itt mutatjuk. A 0-át a körbefutó (a, b, c, d, e, f) szegmensekkel kapjuk, de az '1'-hez csak a (b, c) szegmenseket használjuk. (Az ilyen ábrázolás ötlete Wood nevéhez fűződik [65]. Amúgy Wood eredeti ábrázolása nyolc szegmenst használt, mivel a '4' ábrázolásához a diagonális vonalra szüksége volt.) A hét szegmenses ábrázolás általában a decimális '0', '1', …, '9' számjegyekre alkalmas, de a számítógép-kutatók által használt digitális óráknak a hexadecimális számjegyeket is mutatniuk kell. Így a hét szegmenses logikában megtervezték a tizenhat számjegy ábrázolását,

, (41)

ha adottak rendre az $x_1x_2x_3x_4 = 0000, 0001, 0010, \ldots, 1111$ bemenetek.

Más szóval ki kell értékelnünk az alábbi igazságtáblákkal rendelkező hét Boole-függvényt:

$$\begin{aligned} a &= 1011\ 0111\ 1110\ 0011\,, \\ b &= 1111\ 1001\ 1110\ 0100\,, \\ c &= 1101\ 1111\ 1111\ 0100\,, \\ d &= 1011\ 0110\ 1101\ 1110\,, \\ e &= 1010\ 0010\ 1011\ 1111\,, \\ f &= 1000\ 1111\ 1111\ 0011\,, \\ g &= 0011\ 1110\ 1111\ 1111\,. \end{aligned} \tag{42}$$

Ha külön-külön szeretnénk mindegyik függvényt kiértékelni, akkor néhány eddig tárgyalt módszer alapján belátható, hogy a következő minimális költségek adódnak. $C(a) = 5$, $C(b) = C(c) = C(d) = 6$, $C(e) = C(f) = 5$, és $C(g) = 4$. A hét függvény együttes költsége tehát 37. Mi azonban egyetlen Boole-lánccal együttesen akarjuk mindet megadni. Ezek közül a legrövidebb lesz a leghatékonyabb. Hogyan lehet ezt megtalálni?

Egy valóban optimális lánc megtalálása $\{a, b, c, d, e, f, g\}$-re kiszámítási szempontból megvalósíthatatlannak tűnik. De a korábban bemutatott „lábnyom" ötlet alapján jó megoldást lehet találni. Mi nemcsak egy függvényre tudjuk kiszámítani a minimális költséget, hanem az összes olyan első lépések halmazára is, amelyek hozzásegítenek ehhez a minimális költséghez egy normalizált láncban. Például az e függvény költsége 5, de csak akkor, ha a kiértékelésnél a következő műveletek valamelyikével kezdünk:

$$x_5 = x_1 \oplus x_4 \qquad \text{vagy} \qquad x_5 = x_2 \wedge \bar{x}_3 \qquad \text{vagy} \qquad x_5 = x_2 \vee x_3\,.$$

Szerencsére a kívánt első lépések egyike a hét lábnyom közül négyhez hozzátartozik. A c, d, f, és g mindegyike optimálisan kiértékelhető az $x_5 = x_2 \oplus x_3$ kezdéssel. Így ez egy természetes választás; ez három lépéssel való csökkenést jelent; mint tudjuk, az eredeti 37 lépésből legfeljebb 33 szükséges a befejezéshez.

Most újraszámolhatjuk a költségeket és a lábnyomokat az összes 2^{16} függvényre ugyanúgy, mint az előbb, beállítva az új x_5 függvény költségét nullára. Eredményként a c, d, f, és g költsége 1-el csökken és a lábnyomok is változnak. Például az a függvény költsége 5, de a lábnyoma $\{x_1 \oplus x_3, x_2 \wedge x_3\}$ helyett $\{x_1 \oplus x_3, x_1 \wedge x_4, \bar{x}_1 \wedge x_4, x_2 \wedge x_3, \bar{x}_2 \wedge x_4, x_2 \oplus x_4,$

$x_4 \wedge x_5, x_4 \oplus x_5\}$ lett, amikor az $x_5 = x_2 \oplus x_3$ függvény szabadon elérhetővé vált.

Tény, hogy $x_6 = \bar{x}_1 \wedge x_4$ közös az új lábnyomok közül néggyel, így adódott egy természetes út a folytatásra. Ha mindent újraszámolunk, úgy hogy nulla költséget adunk mind x_5, mind x_6-nak, a következő $x_7 = x_3 \wedge \bar{x}_6$ lépés a legújabb lábnyomok közül ötben közös. Folytatva ezt a „mohó" módszert, nem leszünk mindig ilyen szerencsések, de egy mindössze 22 lépéses lánc felbukkan:

$$\begin{array}{lll} x_5 = x_2 \oplus x_3, & x_{13} = x_1 \oplus x_7, & \bar{a} = x_{20} = x_{14} \wedge \bar{x}_{19}\,, \\ x_6 = \bar{x}_1 \wedge x_4, & x_{14} = x_5 \oplus x_6, & \bar{b} = x_{21} = x_7 \oplus x_{12}\,, \\ x_7 = x_3 \wedge \bar{x}_6, & x_{15} = x_7 \vee x_{12}, & \bar{c} = x_{22} = \bar{x}_8 \wedge x_{15}\,, \\ x_8 = x_1 \oplus x_2, & x_{16} = x_1 \vee x_5, & \bar{d} = x_{23} = x_9 \wedge \bar{x}_{13}\,, \\ x_9 = x_4 \oplus x_5, & x_{17} = x_5 \vee x_6, & \bar{e} = x_{24} = x_6 \vee x_{18}\,, \\ x_{10} = \bar{x}_7 \wedge x_8, & x_{18} = x_9 \wedge x_{10}, & \bar{f} = x_{25} = \bar{x}_8 \wedge x_{17}\,, \\ x_{11} = x_9 \oplus x_{10}, & x_{19} = x_3 \wedge x_9, & g = x_{26} = x_7 \vee x_{16}\,. \\ x_{12} = x_5 \wedge x_{11}\,. & & \end{array} \tag{43}$$

(Ez egy *normalizált* lánc, így az $\{\bar{a}, \bar{b}, \bar{c}, \bar{d}, \bar{e}, \bar{f}, g\}$ normalizáltjait tartalmazza a $\{a, b, c, d, e, f, g\}$ helyett) Egyszerű cserékkel előállhatnak nemnormalizált függvények költségváltozás nélkül.

Parciális függvények. A gyakorlatban egy Boole-függvény kimeneti értékét sokszor csak bizonyos $x_1 \ldots x_n$ bemenetekre definiálják, a többi esetben a kimenetek nem fontosak. Tudhatjuk például, hogy bizonyos bemeneti kombinációk soha sem fordulhatnak elő. Az ilyen esetekben az igazságtábla megfelelő pozíciójába a 0 vagy 1 helyett egy csillagot teszünk.

A hét szegmenses ábrázolás ad egy ilyen esetet, mivel a legtöbb alkalmazás csak azt a tíz binárisan kódolt decimális számot használja inputként, amelyekre $(x_1x_2x_3x_4)_2 \leq 9$. Azzal nem foglalkozunk, hogy mely szegmensek láthatók a többi hat esetben. Így a (42)-ben lévő igazságtábla helyett a következőt használjuk:

$$\begin{aligned} a &= 1011\ 0111\ 11{*}{*}\ {*}{*}{*}{*}, \\ b &= 1111\ 1001\ 11{*}{*}\ {*}{*}{*}{*}, \\ c &= 1101\ 1111\ 11{*}{*}\ {*}{*}{*}{*}, \\ d &= 1011\ 0110\ 11{*}{*}\ {*}{*}{*}{*}, \\ e &= 1010\ 0010\ 10{*}{*}\ {*}{*}{*}{*}, \\ f &= 1000\ 111{*}\ 11{*}{*}\ {*}{*}{*}{*}, \\ g &= 0011\ 1110\ 11{*}{*}\ {*}{*}{*}{*}. \end{aligned} \tag{44}$$

(f-ben az $x_1x_2x_3x_4 = 0111$ pozíción azért van csillag, mivel a '7'-et meg lehet jeleníteni 7 vagy 7 alakban is.) A fejezet írása idején még mindkét változat szinte egyenlő arányban használatos volt. A 6 és a 9 csonkított formája csak régebben volt használatos, de ezek szerencsére már eltűntek.

A csillagokat az igazságtáblában úgy tekintjük mint *érdektelen jeleket* – („don't-cares", ami egy olyan különös kifejezés, amit a villamosmérnökök találtak ki. A 7.8. táblázat mutatja, hogy a kimenetek önkényes megválasztásának szabadsága előnyös. Például $\binom{16}{3}2^{13} = 4{,}587{,}520$ olyan igazságtábla létezik, amely 3 „don't care"-t tartalmaz és ezek 69%-ának a költsége 4 vagy kisebb, míg a csillagmentes igazságtáblák 21%-a tesz lehetővé ilyen gazdaságosságot. Másrészről az érdektelen jelek nem segítenek annyit, mint remélnénk; a 63.

7.8. táblázat.
d ÉRDEKTELEN JELET TARTALMAZÓ,
c KÖLTSÉGŰ 4 VÁLTOZÓS FÜGGVÉNYEK SZÁMA

	$c=0$	$c=1$	$c=2$	$c=3$	$c=4$	$c=5$	$c=6$	$c=7$
$d=0$	10	60	456	2474	10624	24184	25008	2720
$d=1$	160	960	7296	35040	131904	227296	119072	2560
$d=2$	1200	7200	52736	221840	700512	816448	166144	
$d=3$	5600	33600	228992	831232	2045952	1381952	60192	
$d=4$	18200	108816	666528	2034408	3505344	1118128	3296	
$d=5$	43680	257472	1367776	3351488	3491648	433568	32	
$d=6$	80080	455616	2015072	3648608	1914800	86016		
$d=7$	114400	606944	2115648	2474688	533568	12032		
$d=8$	128660	604756	1528808	960080	71520	896		
$d=9$	114080	440960	707488	197632	4160			
$d=10$	78960	224144	189248	20160				
$d=11$	41440	72064	25472	800				
$d=12$	15480	12360	1280					
$d=13$	3680	800						
$d=14$	480							
$d=15$	32							
$d=16$	1							

gyakorlat mutatja, hogy egy tetszőleges függvény, amelynél az igazságtábla 30%-a csillag, a teljesen specifikált függvény költségét körülbelül 30%-kal enyhíti.

Mi az a legrövidebb Boole-lánc, ami a (44)-ben lévő hét parciálisan definiált függvényt kiértékeli? A mi mohó- lábnyom módszerünk könnyen adaptálható a „don't care"-k jelenlétére, mivel OR-ral összekapcsolhajuk annak a 2^d db. függvénynek a lábnyomát, amelyek a d csillaggal összeegyeztethetők. Az egyes függvények külön-külön kiértékelésének kezdeti költségei $C(a) = 3$, $C(b) = C(c) = 2$, $C(d) = 5$, $C(e) = 2$, $C(f) = 3$, $C(g) = 4$-ra redukálódnak, ez összességében 21 a korábbi 31 helyett. A g függvény nem lett olcsóbb, de a lábnyoma nagyobb lett. Ha úgy járunk el mint korábban, miközben a „don't care"-eket előnyben részesítjük, egy olyan alkalmas 13–hosszúságú, láncot találunk, amelyben kimenetenként(!) kevesebb, mint két művelet szerepel.

$$\begin{aligned}
&x_5 = x_1 \oplus x_2, && \bar{e} = x_{10} = x_4 \vee x_8, && \bar{b} = x_{15} = x_2 \wedge \bar{x}_{13}\,,\\
&x_6 = x_3 \wedge \bar{x}_4, && g = x_{11} = x_7 \oplus x_8, && \bar{c} = x_{16} = \bar{x}_2 \wedge x_6,\\
&x_7 = x_1 \oplus x_3, && x_{12} = x_4 \oplus x_{11}, && \bar{f} = x_{17} = \bar{x}_5 \wedge x_9\,.\\
&x_8 = x_2 \wedge \bar{x}_6, && \bar{d} = x_{13} = x_{10} \wedge x_{12}, &&\\
&x_9 = x_3 \vee x_4\,, && \bar{a} = x_{14} = \bar{x}_3 \wedge x_{13}\,, &&
\end{aligned} \tag{45}$$

Amőba játék. Vegyünk most egy népszerű gyermekjátékon alapuló, kissé bővebb problémát. Két játékos egy 3×3 rácsot akar kitölteni. Az egyik egy X-et, a másik egy O-t ír felváltva a rácsba. Addig folytatódik a játék, amíg vagy három X vagy három O megjelenik egy vonalban (Ebben az esetben az a játékos győz), vagy mind a kilenc cella ki van töltve és nincs győztes (ebben az esetben „ablakban a pénz”, azaz döntetlen az eredmény). Például egy játékmenet:

 (46)

Az × győzött. Célunk olyan, az amőba játékot optimálisan játszó gép tervezése, amely minden olyan pozícióban, ahonnan lehetséges a győzelem elérése, győztes lépést lép, és amely nem lép vesztes lépést olyan pozícióban, ahol a vereség elkerülhető.

Pontosabban, az aktuális pozíció celláit megvilágító lámpák vezérlésére a 18 darab $x_1, \ldots, x_9, o_1, \ldots, o_9$ Boole-változó szolgál. A cellák  számozása megegyezik a telefon billentyűzetével. A j cella egy ×-et jelenít meg, ha $x_j = 1$ és O-t, ha $o_j = 1$, illetve sötét marad, ha $x_j = o_j = 0$.[1] Az sohasem állhat elő, hogy $x_j = o_j = 1$, mivel ekkor ‘⊗’-t kellene kijelezni. Feltesszük, hogy az $x_1 \ldots x_9 o_1 \ldots o_9$ változók beállítása egy legális állapotot jelent, amelyben nincs nyertes. A számítógép az ×-et használja és a számítógép léphet. Erre a célra definiáljunk kilenc $y_1, \ldots, y_9$ függvényt, ahol y_j jelentése „változtasd” x_j értékét 0-ról 1-re. Ha az aktuális pozíció „döntetlen”, akkor kötelező az $y_1 = \cdots = y_9 = 0$ beállítás, mert egyébként pontosan egy y_j értéknek kell 1-nek lennie, de y_j értéke csak akkor lehet 1, ha $x_j = o_j = 0$.

> Elkezdtem egy játék vizsgálatát, a neve „tit-tat-to” . . .
> célom kideríteni, hogy milyen kombinációszám lenne szükséges
> a lépések és a helyzetek összes lehetséges változatára.
> Azt gondoltam, hogy ennek viszonylag nincs jelentősége.
> . . . Azonban egy szokatlan nehézség merült fel.
> Ha az automatának lépni kell, előfordulhat, hogy
> két különböző lépés adódik és mindegyik egyformán biztosítja neki a győzelmet.
> . . . Hacsak valamilyen óvintézkedést nem vezetünk be,
> a gép megkísérelhet két egymásnak ellentmondó mozgást.
> — CHARLES BABBAGE, *Passages from the Life of a Philosopher*
> (1864) [6, 34. fejezet, 154–155. oldal]

Tizennyolc változóval a mi kilenc y_j függvényünk igazságtáblája $2^{18} = 262{,}144$ méretű. Előfordul, hogy $x_1 \ldots x_9 o_1 \ldots o_9$-ra csak 4520 legális bemenet lehetséges, így aztán ezeknek az igazságtábláknak 98.3%-a érdektelen jelekkel lesz kitöltve. Mivel a 4520 kellemetlenül nagy az értelmező Boole-lánc intuitív megértéséhez és megtervezéséhez. A 7.1.4. pontban a Boole-függvények reprezentálására olyan alternatív módokat mutatunk, amelyekkel gyakran lehetséges változók százaival foglalkozni, még akkor is, ha a kezelendő igazságtábla lehetetlenül nagy [5].

A legtöbb 18 változós függvényhez több, mint $2^{18}/18$ kapura van szükség, de reménykedjünk, hogy tudjuk ezt jobban is csinálni. A amőba játékban egy megfelelő lépés megtételére közvetlenül adódik egy kézenfekvő stratégia, néhány könnyen felismerhető feltétel

[1] E beállítás alapja egy a kora 1950-es években Csikágóban a Museum of Science and Industry-ban megrendezett kiállítás volt, ahol a szerző először lett bevezetve a kapcsoló áramkörök csodáiba. A csikágói gép, amit nagyjából 1940-ben tervezett Keister a Bell Telephone Laboratories-ban [24], megengedte, hogy elsőként lépjek. Gyorsan megértettem, hogy a gépet nem lehet megverni. Ekkor elhatároztam, hogy a lehető legostobább lépéseket teszem meg, abban a reményben, hogy a tervező nem készült fel ilyen bizarr viselkedésre. Lényegében elértem, hogy a gép olyan pozícióba kerüljön, ahonnan két győztes lépést tehetett. A gép mindkettőt végrehajtotta. Kétszer lépni a szabályok kirívó megsértése, ezért ez számomra erkölcsi győzelem volt még akkor is, ha a gép azt közölte, hogy vesztettem.

formájában:

w_j,	× beírása a	j cellába győzelem, egy ×-eket tartalmazó sor befejezése miatt;
b_j,	O beírása a	j cellába veszteség lenne, egy O-ás sor befejezése miatt;
f_j,	× beírása a	j cellába az × számára két esélyt ad a győzelemre;
d_j,	O beírása a	j cellába a O számára két esélyt adna a győzelemre.

Például a (46)-ban × középre lép, mert a O-t blokkolnia kell. Ezért ez egy b_5-típusú és egyben f_5-típusú lépés, ami győzelmet biztosít a következő lépésben.

Legyenek $L = \{\{1,2,3\},\{4,5,6\},\{7,8,9\},\{1,4,7\},\{2,5,8\},\{3,6,9\},\{1,5,9\},\{3,5,7\}\}$ a győztes sorok. Ekkor

$$m_j = \bar{x}_j \wedge \bar{o}_j; \qquad \text{[lépés a } j\text{-edik cellába legális]} \tag{47}$$

$$w_j = m_j \wedge \textstyle\bigvee_{\{i,j,k\}\in L}(x_i \wedge x_k); \qquad \text{[lépés a } j\text{-edik cellába győzelem]} \tag{48}$$

$$b_j = m_j \wedge \textstyle\bigvee_{\{i,j,k\}\in L}(o_i \wedge o_k); \qquad \text{[lépés a } j\text{-edik cellába blokkol]} \tag{49}$$

$$f_j = m_j \wedge S_2\bigl(\{\alpha_{ik} \mid \{i,j,k\}\in L\}\bigr); \qquad \text{[lépés a } j\text{-edik cellába közrefog]} \tag{50}$$

$$d_j = m_j \wedge S_2\bigl(\{\beta_{ik} \mid \{i,j,k\}\in L\}\bigr); \qquad \text{[moving in cell } j \text{ védekezik]} \tag{51}$$

itt α_{ik} és β_{ik} egy ×-et vagy O-át jelöl egy blank-kel együtt, nevezetesen

$$\alpha_{ik} = (x_i \wedge m_k) \vee (m_i \wedge x_k), \qquad \beta_{ik} = (o_i \wedge m_k) \vee (m_i \wedge o_k). \tag{52}$$

Például, $b_1 = m_1 \wedge \bigl((o_2 \wedge o_3) \vee (o_4 \wedge o_7) \vee (o_5 \wedge o_9)\bigr)$; $f_2 = m_2 \wedge S_2(\alpha_{13}, \alpha_{58}) = m_2 \wedge \alpha_{13} \wedge \alpha_{58}$; $d_5 = m_5 \wedge S_2(\beta_{19}, \beta_{28}, \beta_{37}, \beta_{46})$. Ezekkel a definíciókkal megpróbálhatjuk a lépéseket sorba rendezni:

$$\{w_1,\ldots,w_9\} > \{b_1,\ldots,b_9\} > \{f_1,\ldots,f_9\} > \{d_1,\ldots,d_9\} > \{m_1,\ldots,m_9\}. \tag{53}$$

„Győzz, ha lehet, egyébként blokkolj, ha lehet, egyébként csinálj befogást, ha lehet, egyébként védekezz, ha lehet, egyébként lépj meg egy legális lépést.” A későbbiekben, amikor a legális lépések közül választunk, ésszerűnek látszik a következő rendezés:

$$m_5 > m_1 > m_3 > m_9 > m_7 > m_2 > m_6 > m_8 > m_4, \tag{54}$$

Ugyanis az 5, a középső cella négy győztes sorban jelenik meg. Ha az 1, 3, 9, vagy 7 sarokcellába lépünk, akkor háromban, ha a 2, 6, 8, vagy 4 oldalcellába lépünk, akkor már csak kettőben van benne. Ezt a lépésrendezést az (53)-beli $\{w_j\}$, $\{b_j\}$, $\{f_j\}$, $\{d_j\}$, and $\{m_j\}$ lépéscsoportoknál is használhatjuk.

Annak biztosítására, hogy egyszerre csak egy lépés legyen kiválasztva definiáljuk a w'_j, b'_j, f'_j, d'_j, m'_j-t, azt sugallva, hogy „az első választás a legjobb”. Ennek megfelelően $w'_5 = 0$, $w'_1 = w_5$, $w'_3 = w_1 \vee w'_1$, …, $w'_4 = w_8 \vee w'_8$, $b'_5 = w_4 \vee w'_4$, $b'_1 = b_5 \vee b'_5$, …, $m'_4 = m_8 \vee m'_8$. A amőba automata definícióját kiegészíthetjük a következővel:

$$y_j = (w_j \wedge \bar{w}'_j) \vee (b_j \wedge \bar{b}'_j) \vee (f_j \wedge \bar{f}'_j) \vee (d_j \wedge \bar{d}'_j) \vee (m_j \wedge \bar{m}'_j) \quad (1 \le j \le 9). \tag{55}$$

Így konstruáltunk az m-ekre 9 kaput, a w-kre 48-at, a b-kre 48-at, az α-kra és a β-kra 144-et, az f-ekre 35-öt (a 7.9. ábra segítségével), a d-kre 35-öt, a prím változókra 43-at és az y-okra 80-at. A továbbiakban a négyváltozós parciális függvényekkel kapcsolatos tudásunkat felhasználhatjuk arra, hogy az (52)-beli hat műveletet négyre csökkentsük.

$$\alpha_{ik} = (x_i \oplus x_k) \vee \overline{(o_i \oplus o_k)}, \qquad \beta_{ik} = \overline{(x_i \oplus x_k)} \vee (o_i \oplus o_k)\,. \tag{56}$$

Ezzel a trükkel 48 kaput nyerünk, így a teljes költség 396 kapu összesen.

A amőba játéknak a (47)–(56)-ban kifejtett stratégiája a legtöbb esetben jól működik, de van néhány kirívó hibája. Például szégyellni valóan veszít a következő játékban

 ; (57)

a második × lépés d_3, védekezésként az ○ befogására, arra késztet ○ játékost, hogy lépjen az ellentétes sarokba! Egy másik hiba is felmerül. Például a  pozíció után, amikor az m_5 lépés a  vesztéshez vezet, pedig × győzhetett volna, lásd (46). A 65. gyakorlat javít a dolgon és egy teljesen korrekt Boole amőba-játékost ad meg, amihez pontosan 445 kapura van szükség.

Függvény dekompozíció. Ha az $f(x_1,\ldots,x_n)$ függvényt fel lehet írni $g(x_1,\ldots,x_k, h(x_{k+1},\ldots,x_n))$ alakban, akkor a kiszámításhoz jó ötlet először az $y = h(x_{k+1},\ldots,x_n)$-t kiértékelni és azután kiszámítani $g(x_1,\ldots,x_k,y)$-t. Az ilyen dekompozíciók tanulmányozását Ashenhurst kezdeményezte 1952-ben, lásd [4]. Azt is megfigyelte, hogy elég könnyű felismerni, ha f rendelkezik ezzel a speciális tulajdonsággal. Írjuk le az f igazságtábláját $2^k \times 2^{n-k}$ mátrixszal, mint (36)-ban, ahol a sorok az $x_1 \ldots x_k$-nak, az oszlopok pedig az $x_{k+1} \ldots x_n$-nek vannak megfeleltetve. A kívánt g és h részfüggvények akkor és csak akkor léteznek, ha ennek a mátrixnak az oszlopai legfeljebb kétfélék. Például a $\langle x_1 x_2 \langle x_3 x_4 x_5\rangle\rangle$ függvény igazságtáblája

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1, \end{matrix}$$

ha ebbe a kétdimenziós formába írjuk. Az egyik fajta oszlop megfelel a $h(x_{k+1},\ldots,x_n) = 0$-nak, a másik fajta pedig a $h(x_{k+1},\ldots,x_n) = 1$-nek.

Általában az $X = \{x_1,\ldots,x_n\}$ változókat bármely két $Y = \{y_1,\ldots,y_k\}$ és $Z = \{z_1,\ldots,z_{n-k}\}$ diszjunkt részhalmazra bonthatjuk. Ennek alapján megkaphatjuk az $f(x) = g(y,h(z))$-et. Egy adott (Y,Z) dekompozícióra meg kell vizsgálni a $2^k \times 2^{n-k}$ igazságtábla oszlopait, ahol a sorok felelnek meg az y értékeinek. Azonban X-re 2^n partíció létezik és ezek mindegyike potenciálisan alkalmas lehet, kivéve olyan triviális eseteket, mint $|Y| = 0$ vagy $|Z| \le 1$. Hogyan kerülhetjük el ezt az óriási számú vizsgálatot?

Egy gyakorlati lehetőség Shen, McKellar és Weiner nevéhez fűződik [50]. Ezzel a módszerrel a potenciálisan hasznos (Y,Z) partíciók azonosításához csak $O(n^2)$ lépés szükséges. Az alapötlet egyszerű; tegyük fel hogy $x_i \in Z$, $x_j \in Z$, és $x_m \in Y$. definiáljunk nyolc δ_l bináris vektort $l = (l_1 l_2 l_3)_2$-ra, ahol δ_l-beli (i,j,m) komponensekben rendre (l_1,l_2,l_3) van és nulla máshol. Tekintsünk bármilyen véletlenül választott $x = x_1 \ldots x_n$ vektort és értékeljük ki $f_l = f(x \oplus \delta_l)$-t $0 \le l \le 7$-re. Ekkor az

$$\binom{f_0}{f_1} \qquad \binom{f_2}{f_3} \qquad \binom{f_4}{f_5} \qquad \binom{f_6}{f_7} \tag{58}$$

párok jelennek meg egy $2^k \times 2^{n-k}$ igazságtábla 2×4-es részmátrixában. A dekompozíció lehetetlen, ha ezek a párok különböznek, vagy ha három különböző értéket tartalmaznak.

Nevezzük a párokat „jónak” , ha egyformák vagy ha csak két különböző értékük van. Egyébként ezek „rosszak”. Ha f lényegében véletlenszerű viselkedésű, akkor – a fenti ötletet néhány véletlenszerűen kiválasztott x vektorra alkalmazva – hamar találunk rossz párokat. Ugyanis az $f_0 f_1 \ldots f_7$-re lehetséges 256 esetből csak 88 ad jó párhalmazt. Annak a valószínűsége, hogy egy sorban tízszer találjunk jó párokat, csak $(\frac{88}{256})^{10} \approx .00002$. Ha rossz párokat találunk, akkor megállapíthatjuk, hogy

$$x_i \in Z \quad \text{és} \quad x_j \in Z \implies x_m \in Z, \tag{59}$$

mivel az $x_m \in Y$ alternatív lehetőség lehetetlen.

Tegyük fel például, hogy $n = 9$ és, hogy az f függvény igazságtáblája 11001001000011 $\ldots$00101 a π legfontosabb 512 bitjéből áll. (Ez a „több-vagy-kevesebb véletlen függvény”, amit $n = 4$-re tanulmányoztunk fentebb a (5)-ben és a (6)-ban) Erre a π függvényre a rossz párok gyorsan megtalálhatók minden olyan (i, j, m) esetre, ahol $m \neq i < j \neq m$. A szerző tapasztalata, hogy a 252 olyan esetből, ahol a véletlen x vektorok esetenkénti arányszáma 1.52, 170-et azonnal meghatároztak. Csak egy esetben volt szükség mind a nyolc x-re, mielőtt a rossz párok megjelentek. Az (59) minden releváns (i, j, m)-re teljesül ezért világos, hogy a függvény dekomponálhatatlan. A 73. gyakorlat mutatja, hogy nincs szükség mind a 252 teszt elvégzésére. Ugyanis ahhoz, hogy a π függvény dekomponálhatatlanságát igazoljuk; elegendő mindössze $\binom{n}{2} = 36$ teszt.

Tekintsünk egy kisebb véletlen függvényt. Legyen $f(x_1, \ldots, x_9) = (\det X) \mod 2$, ahol

$$X = \begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix}. \tag{60}$$

Ez a függvény nem teljesíti az (59)-t, ha $i = 1$, $j = 2$, és $m = 3$, mivel ezekben az esetekben nincsenek rossz párok. Azonban $4 \leq m \leq 9$-re, ha $\{i, j\} = \{1, 2\}$ teljesíti a (59)-t. Ezt a viselkedést az ‘12 $\Rightarrow$ 456789’ rövidítéssel jelöljük. Az implikációk teljes halmaza minden $\{i, j\}$ párra a következő:

12 ⇒ 456789	18 ⇒ 34569	27 ⇒ 34569	37 ⇒ 24568	48 ⇒ 12369	67 ⇒ 12358
13 ⇒ 456789	19 ⇒ 24568	28 ⇒ 134679	38 ⇒ 14567	49 ⇒ 12358	68 ⇒ 12347
14 ⇒ 235689	23 ⇒ 456789	29 ⇒ 14567	39 ⇒ 124578	56 ⇒ 123789	69 ⇒ 124578
15 ⇒ 36789	24 ⇒ 36789	34 ⇒ 25789	45 ⇒ 123789	57 ⇒ 12369	78 ⇒ 123456
16 ⇒ 25789	25 ⇒ 134679	35 ⇒ 14789	46 ⇒ 123789	58 ⇒ 134679	79 ⇒ 123456
17 ⇒ 235689	26 ⇒ 14789	36 ⇒ 124578	47 ⇒ 235689	59 ⇒ 12347	89 ⇒ 123456

(lásd a 69. gyakorlatot). Kissé nehezebb rossz párokat találni, ha találomra választjuk a függvényt. A szerző tapasztalata szerint, ha léteztek rossz párok, akkor a szükséges x-ek átlaga 3.6 körülire nőtt. Természetesen a tesztszámot korlátozni kell egy t küszöbszámmal, ami azt jelenti, hogy leállunk, ha t egymás utáni próbálkozásra sem találtunk rossz párt. $t = 10$ választás mellett a fent kilistázott 198 implikációból mintegy 8-at meg kell találni.

Az (59)-hez hasonló implikációk a Horn-klózok és a 7.1.1. pontból tudjuk, hogy Horn-klózokból könnyű további levezetéseket előállítani. A 74. gyakorlatban látható módszerrel le tudjuk vezetni, hogy $|Z| > 1$ esetben az egyetlen lehetséges partíció a triviális ($Y = \emptyset$, $Z = \{x_1, \ldots, x_9\}$). Ehhez elég volt 50-nél kevesebb (i, j, m) eset megvizsgálása.

Hasonló eredményeket kapunk, ha $f(x_1, \ldots, x_9) = [\mathrm{per}\, X > 0]$, ahol a per a *permanens* függvényt jelöli. (Ebben az esetben f megadja, hogy a $K_{3,3}$ páros részgráfjában van-e az $x_1 \ldots x_9$ változók által meghatározott élekkel illesztés.) Itt 180 implikációt láthatunk,

$12 \Rightarrow 456789$	$18 \Rightarrow 3459$	$27 \Rightarrow 3459$	$37 \Rightarrow 2468$	$48 \Rightarrow 1269$	$67 \Rightarrow 1358$
$13 \Rightarrow 456789$	$19 \Rightarrow 2468$	$28 \Rightarrow 134679$	$38 \Rightarrow 1567$	$49 \Rightarrow 1358$	$68 \Rightarrow 2347$
$14 \Rightarrow 235689$	$23 \Rightarrow 456789$	$29 \Rightarrow 1567$	$39 \Rightarrow 124578$	$56 \Rightarrow 123789$	$69 \Rightarrow 124578$
$15 \Rightarrow 3678$	$24 \Rightarrow 3678$	$34 \Rightarrow 2579$	$45 \Rightarrow 123789$	$57 \Rightarrow 1269$	$78 \Rightarrow 123456$
$16 \Rightarrow 2579$	$25 \Rightarrow 134679$	$35 \Rightarrow 1489$	$46 \Rightarrow 123789$	$58 \Rightarrow 134679$	$79 \Rightarrow 123456$
$17 \Rightarrow 235689$	$26 \Rightarrow 1489$	$36 \Rightarrow 124578$	$47 \Rightarrow 235689$	$59 \Rightarrow 2347$	$89 \Rightarrow 123456$,

,

amelyekből csak 122-t lehetett volna megtalálni a $t = 10$ vágási küszöbbel. (A legjobb választás t-re nem ismert; talán dinamikusan változik) Mégis ez a 122 Horn-klóz több mint elegendő a dekomponálhatatlanság igazolására.

Mit mondhatunk a dekomponálható függvényről? $f = \langle x_2 x_3 x_6 x_9 \langle x_1 x_4 x_5 x_7 x_8 \rangle\rangle$ estén azt kapjuk, hogy $i \wedge j \Rightarrow m$ minden $m \notin \{i, j\}$-re, kivéve ha $\{i, j\} \subseteq \{1, 4, 5, 7, 8\}$. Ez utóbbi esetben m-nek az $\{1, 4, 5, 7, 8\}$-hoz kell tartoznia. Jóllehet a 212 implikációból csak 185-öt találtak meg $t = 10$-re, az $Y = \{x_2, x_3, x_6, x_9\}$, $Z = \{x_1, x_4, x_5, x_7, x_8\}$ partíció hamar felbukkant, mint erős lehetőség.

Amikor egy potenciális dekompozíció ötletszerűen felmerül, mindig meg kell vizsgálni a megfelelő $2^k \times 2^{n-k}$ igazságtáblát, hogy valóban csak egy- vagy kétféle oszlopai vannak-e. Szívesen használunk fel 2^n időegységet erre a vizsgálatra, mert így nagymértékben egyszerűsítjük az f kiértékelését.

Az $f = [(x_1 x_2 x_3 x_4)_2 \geq (x_5 x_6 x_7 x_8)_2 + x_9]$ összehasonlító függvény egy másik érdekes eset. Ennek 184 potenciálisan levezethető implikációja:

$12 \Rightarrow 3456789$	$18 \Rightarrow 2345679$	$27 \Rightarrow 34689$	$37 \Rightarrow 489$	$48 \Rightarrow 9$	$67 \Rightarrow 23489$
$13 \Rightarrow 2456789$	$19 \Rightarrow 2345678$	$28 \Rightarrow 34679$	$38 \Rightarrow 479$	$49 \Rightarrow 8$	$68 \Rightarrow 23479$
$14 \Rightarrow 2356789$	$23 \Rightarrow 46789$	$29 \Rightarrow 34678$	$39 \Rightarrow 478$	$56 \Rightarrow 1234789$	$69 \Rightarrow 23478$
$15 \Rightarrow 2346789$	$24 \Rightarrow 36789$	$34 \Rightarrow 789$	$45 \Rightarrow 1236789$	$57 \Rightarrow 1234689$	$78 \Rightarrow 349$
$16 \Rightarrow 2345789$	$25 \Rightarrow 1346789$	$35 \Rightarrow 1246789$	$46 \Rightarrow 23789$	$58 \Rightarrow 1234679$	$79 \Rightarrow 348$
$17 \Rightarrow 2345689$	$26 \Rightarrow 34789$	$36 \Rightarrow 24789$	$47 \Rightarrow 389$	$59 \Rightarrow 1234678$	$89 \Rightarrow 4$,

és ezeek közül 145-öt $t = 10$ mellett találtak meg. Három dekompozíció szinte adta magát, ezeknél rendre $Z = \{x_4, x_8, x_9\}$, $Z = \{x_3, x_4, x_7, x_8, x_9\}$ és $Z = \{x_2, x_3, x_4, x_6, x_7, x_8, x_9\}$. Ashenhurst bizonyította [4], hogy f azonnal redukálható, amikor egy nemtriviális dekompozíció adódik. A többi dekompozíció felmerülhet később, amikor a g és h függvényeket próbáljuk redukálni.

***Parciális függvények dekompozíciója.** Ha az f függvény csak parciálisan specifikált, akkor egy (Y, Z) partíciójú dekompozíció léte attól függ, hogy a érdektelen jelekhez hogyan tudunk úgy értékeket rendelni, hogy legfeljebb két különböző oszlop jelenjen meg a megfelelő $2^k \times 2^{n-k}$ igazságtáblában.

A 0, 1, és $*$ jelekből álló $u_1 \ldots u_m$ és $v_1 \ldots v_m$ vektorokat inkompatibilisnek nevezzük, ha van olyan j, hogy $u_j = 0$ és $v_j = 1$ vagy $u_j = 1$ és $v_j = 0$. Más szóval, ha az u és v által definiált m-kockák részkockáinak nincs közös pontja. Tekintsük azt a gráfot, amelynek a csúcsai az érdektelen jeleket is tartalmazó igazságtábla oszlopai, ahol u—v akkor és csak akkor, ha u és v inkompatibilisek. A $*$-oknak akkor és csak akkor adhatunk úgy értéket, hogy legfeljebb két különböző oszlop álljon elő, ha ez a gráf *páros*. Ha $u_1, \ldots, u_l$ kölcsönösen kompatibilisek, akkor $u_1 \sqcup \cdots \sqcup u_l$, a 7.1.1-32. gyakorlatban definiált általánosított konszenzusuk kompatibilis mindegyikkel. Lásd Hight [22], valamint Boros, Gurvich, Hammer, Ibaraki és Kogan cikkét [10]. Mivel egy gráf akkor és csak akkor páros, ha nem tartalmaz páratlan hosszúságú kört, ez a feltétel egy mélységi kereséssel könnyen tesztelhető (Lásd a 7.4.1. pontot).

Következésképp Shen, McKellar, és Weiner módszere [50] működik akkor is, ha érdektelen értékek is vannak. Az (58)-beli négy pár akkor és csak akkor tekintendő rossznak , ha közülük három kölcsönösen inkompatibilis. Ugyanúgy dolgozhatunk mint korábban, bár a rossz párok megtalálása nehezebb, amikor sok $*$ fordul elő (lásd a 72. gyakorlat). Azonban Ashenhurst tétele [4] már nem alkalmazható. Ha néhány dekompozíció már létezik, akkor az összeset megtalálhatjuk később, mivel az érdektelenekre különböző beállításokat használhatunk és ezek közül bizonyosak jobbak lehetnek, mint mások.

Bár a legtöbb $f(x)$ függvénynek nincs egyszerű $g(y, h(z))$ dekompozíciója, nem szabad túl hamar elveszíteni a reményt, mivel más, például a $g(y, h_1(z), h_2(z))$-hoz hasonló alakkal kaphatunk hatékony láncot. Ha például f szimmetrikus a $\{z_1, z_2, z_3\}$ három változóban, akkor mindig felírható az $f(x) = g\big(y, S_{1,2}(z_1, z_2, z_3), S_{1,3}(z_1, z_2, z_3)\big)$ alak, mivel $S_{1,2}(z_1, z_2, z_3)$ és $S_{1,3}(z_1, z_2, z_3)$ jellemzi az $z_1 + z_2 + z_3$ értékét. (Megjegyezzük, hogy négy lépés pontosan elegendő az $S_{1,2}$ és $S_{1,3}$ kiszámításához.)

Curtis megfigyelése szerint [13] $f(x)$-et akkor és csak akkor lehet kifejezni $g(y, h_1(z), \ldots, h_r(z))$ alakban, ha az Y-nak és Z-nek megfelelő $2^k \times 2^{n-k}$-as igazságtábla legfeljebb 2^r különböző oszlopot tartalmaz. Ha don't-care-ek is találhatók, akkor ugyanez az eredmény akkor és csak akkor érhető el, ha az Y és Z-hez tartozó inkompatibilitási gráf legfeljebb 2^r színnel kiszínezhető.

Például a fentebb vizsgált $f(x) = (\det X) \mod 2$ függvény a $Z = \{x_4, x_5, x_6, x_7, x_8, x_9\}$ esetén nyolc különböző oszlopot eredményez, ami meglepően kis szám, mivel az igazságtáblának 8 sora és 64 oszlopa van. Ebből a tényből eljuthatunk arra a gondolatra, hogy hogyan fejtsük ki a determinánst az első sor kofaktoraival,

$$f(x) \;=\; x_1 \wedge h_1(x_4, \ldots, x_9) \,\oplus\, x_2 \wedge h_2(x_4, \ldots, x_9) \,\oplus\, x_3 \wedge h_3(x_4, \ldots, x_9),$$

ha nem ismernénk már ilyen szabályt.

Ha $d \le 2^r$ különböző oszlop fordul elő, akkor az $f(x)$-et tekinthetjük úgy, mint y és $h(z)$ függvényét, ahol h az egyes $z_1 \ldots z_{n-k}$ bináris vektorokhoz a $\{0, 1, \ldots, d-1\}$ értékek valamelyikét hozzárendeli. Így $(h_1, \ldots, h_r)$ lényegében kódolja a különböző oszloptípusokat. Reméljük, hogy találunk olyan egyszerű $h_1, \ldots, h_r$ függvényeket, amelyek megadnak egy ilyen kódolást. Amennyiben d sokkal kisebb, mint 2^r, akkor a $g(y, h_1, \ldots, h_r)$ függvényben sok don't-care szerepel, ami nagyban csökkenti a költséget.

A különböző oszlopok sugallhatnak olyan g függvényt, amelyhez a h-k don't-care-eket tartalmaznak. Például, ha az oszlopok $(0,0,0,0)^T$ vagy $(0,0,1,1)^T$ vagy $(0,1,1,0)^T$, akkor $g(y_1, y_2, h_1, h_2) = (y_1 \oplus (h_1 \wedge y_2)) \wedge h_2$ alkalmas. Itt, ha z a csupa nulla oszlopnak felel meg, akkor a $h_1(z)$ értéke tetszőleges. Curtis megmutatta, hogy ezt az ötletet hogyan lehet kiaknázni, ha $|Y| = 1$ és $|Z| = n - 1$ (lásd [14]).

A dekompozíciós technikák átfogó tárgyalásához lásd Karp cikkét [23].

n **nagyobb értékei.** Eddig Boole-függvényekre csak kisméretű példákat tekintettünk. Az S tétel szerint a nagy véletlen példák magától értetődően nehezek. Azonban a gyakorlati példák lehetnek nagyon is nem véletlenszerűek. Ezért van értelme annak, hogy az egyszerűsítésekre heurisztikus módszereket keressünk.

n növekedése esetén, a Boole-függvények kezelésére a jelenleg ismert legjobb mód a Boole-lánccal való kezdés – nem egy óriási igazságtáblával – majd ezt a láncot kell tökélesíteni „helyi változtatásokkal”. A lánc megadható egy egyenlethalmazzal. Ha egy közbenső eredményt viszonylag kevés részsorozatban használunk, akkor megpróbálhatjuk ezt kihagyni és időlegesen berakni egy háromváltozós függvénybe, majd újraformalizálni ezeket a függvényeket és ezáltal a láncot javítani, ha lehet.

Például tegyük fel, hogy az $x_i = x_j \circ x_k$ kaput egyetlen egyszer használjuk az $x_l = x_i \square x_m$ kapuhoz, így aztán $x_l = (x_j \circ x_k) \square x_m$. Lehet, hogy már vannak olyan kapuk, amelyekkel x_j, x_k és x_m más függvényeit kiszámítottuk. Viszont az x_j, x_k és x_m definíciói maguk után vonhatják, hogy az (x_j, x_k, x_m)-hez rendelhető bizonyos értékek nem lehetségesek. Ily módon x_l-et kiszámíthatjuk más kapukból, mindössze egy további művelet végrehajtásával. Például, ha $x_i = x_j \wedge x_k$ és $x_l = x_i \vee x_m$, és ha az $x_j \vee x_m$ és $x_k \vee x_m$ értékek megjelennek valahol a láncban, akkor beírhatjuk $x_l = (x_j \vee x_m) \wedge (x_k \vee x_m)$-et; ez kiküszöböli x_i-t és 1-gyel csökkenti a költséget. Vagy, ha mondjuk $x_j \wedge (x_k \oplus x_m)$ előfordul valahol és tudjuk, hogy $x_j x_k x_m \neq 101$, akkor $x_l = x_m \oplus (x_j \wedge (x_k \oplus x_m))$ lehetséges.

Ha x_i-t csak x_l-ben használtuk és x_l-et csak x_p-ben, akkor az x_p kapu négy változótól függ. Ekkor a költséget lehetséges csökkenteni a négyváltozós függvényekkel kapcsolatos teljes tudásunk felhasználásával és x_p-re az x_i és x_l eliminálásával egy jobb megoldást kapni. Hasonlóan, ha x_i csak az x_l-ben és x_p-ben jelenik meg, akkor x_i eliminálható, ha találunk jobb lehetőséget két különböző négyváltozós függvény kiértékelésére, esetleg érdektelen értékekkel és olyan egyéb, négyváltozós függvényekkel, amelyekben a négy változó szabadon elérhető. Másrészt az ilyen problémákat meg tudjuk oldani a fentebb tárgyalt „lábnyom” módszerrel.

Ha lokális cserékkel nem tudjuk csökkenteni a költséget, akkor olyan lokális cserékkel próbálkozunk, amelyek megtartják, esetleg növelik a költséget, miközben olyan láncok alakulnak ki, amelyeket más módon lehet egyszerűsíteni. Az ilyen lokális vizsgálati módszerekkel a 7.10. alfejezetben behatóan foglalkozunk.

A Boole optimalizálási technikákról, vagy ahogy az elektronikai mérnökök mondják a „többszintű logikai szintézis” problémájáról remek összefoglalást publikáltak Brayton, Hachtel és Sangiovanni-Vincentelli a [11] cikkben és De Micheli a könyvében [34].

Alsó korlátok. Az S tétel szerint majdnem minden $2^n/n$-et meghaladó hosszúságú láncot igénylő $n \geq 12$ változós függvényt nehéz kiértékelni. Persze a modern számítógépek, amelyek Boole-változók ezreit reprezentáló, elektronikus jelekre működő logikai áramkörökből vannak felépítve, mikroszekundumonként Boole-függvények millióit értékelik ki. Nyilvánvaló, hogy rengeteg gyorsan kiértékelhető fontos függvény létezik a D tétel ellenére. Ezt a tételt indirekt módon bizonyítottuk; egyszerűen az alacsony költségű eseteket számoltuk, így szinte semmit sem tudtunk meg olyan különleges példákról, amelyek a gyakorlatban merülnek fel. Ha ki akarunk számítani egy adott függvényt és a munkát csak nagyon összetettnek képzelhetjük el, hogyan lehetünk biztosak abban, hogy nincs egy trükkös rövidítés?

A válasz erre a kérdésre botrányos. Több évtizedes intenzív kutatás után a számítástudomány szakemberei képtelenek megtalálni *bármilyen* olyan $f(x_1, \ldots, x_n)$ függvénycsaládot, aminek a költsége az n növekedésével rá jellemzően nemlineáris. A valóságos viselkedés $2^n/n$, de egy $n \lg\lg\lg n$ erősségű alsó határ nincs bizonyítva! Természetesen ki lehet olyan mesterséges példákat találni, mint a „lexikografikusan legkisebb, 2^n hosszúságú igazságtábla, ami nem állítható elő egyetlen $\lfloor 2^n/n \rfloor - 1$ hosszúságú Boole-lánccal sem"; ám az ilyen függvényeket biztosan nem lehet kifejezni. Egy kifejezhető $f(x_1, \ldots, x_n)$ függvény igazságtáblájának kiszámíthatónak kell lennie mondjuk legfeljebb 2^{cn} időegység alatt valamilyen c konstans mellett. Más szóval az összes függvényérték kiszámításához szükséges időnek polinomiálisnak kell lennie az igazságtábla hosszának függvényében. Ezek alapján, jelenleg nem ismert olyan egykimenetű függvénycsalád, aminek a kombinatorikus bonyolultsága meghaladná a $3n + O(1)$ értéket amikor $n \to \infty$. Lásd [9].

A kép azonban nem teljesen sötét, mivel a gyakorlatban fontos függvényekre bizonyítottak egy sor érdekes *lineáris* alsó korlátot. Az ilyen eredmények elérésének alapvető módszereit Redkin vezette be 1970-ben. Tegyük fel, hogy $f(x_1, \ldots, x_n)$-re már van egy r költségű optimális lánc. $x_n \leftarrow 0$ vagy $x_n \leftarrow 1$ beállítás mellett a $g(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 0)$ és $h(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 1)$ függvényekre redukált láncot kapunk, amelynek $r - u$ a költsége, ha u különböző kapunak x_n a bemenete. Ezenkívül, ha x_n-et egy olyan $x_i = x_n \circ x_k$ „irányító" kapuban használtuk fel, ahol a $\circ$ művelet sem nem az $\oplus$ sem nem az $\equiv$, akkor az x_n bizonyos beállítása kényszeríti, hogy x_i konstans legyen, ami további redukciót eredményez g vagy h láncán. A g és/vagy h alsó korlátai így az f alsó korlátját meghatározzák. (Lásd a 77–81. gyakorlatokat.)

De mi van a nemlineáris alsó korlátok bizonyításával? Majdnem minden probléma, amire a válasz igen-nem, leírható egy Boole-függvénnyel, ezért nincs hiány olyan explicit függvényekben, amelyeket nem tudunk kiértékelni lineáris időben, de még polinomiális időben sem. Például egy $\{v_1, \ldots, v_m\}$ a csúcsokkal rendelkező G irányított gráf, leírható az X átmenet mátrixszal, ahol $x_{ij} = [v_i \to v_j]$:

$$f(x_{12}, \ldots, x_{1m}, \ldots, x_{m1}, \ldots, x_{m(m-1)}) = [G \text{ -nek van Hamilton-útja}] \qquad (61)$$

egy $n = m(m-1)$ változós Boole-függvény. Nagyon jó lenne, ha ezt a függvényt ki tudnánk értékelni, mondjuk n^4 lépésben. Azt tudjuk, hogyan számítsuk ki f igazságtábláját $O(m!2^n) = 2^{n+O(\sqrt{n}\lg n)}$ lépésben. Mivel csak $m!$ potenciális Hamilton út létezik, az f „explicit". Azt azonban senki sem tudja, hogyan értékeljük ki f-et polimomiális időben, vagy hogyan bizonyítsuk, hogy $4n$-lépéses lánca nem létezik.

Általában azt tudjuk, hogy minden n esetén f-re létezhetnek rövid Boole-láncok. Mindenesetre 7.9. és a 7.10. ábra mutat lehetőségeket döbbenetesen ügyes láncokra még a 4 és 5 változós esetben is. A nagyméretű, valaha is megoldásra váró problémákkal kapcsolatos hatékony láncok vizsgálata még a „jövő zenéje" – látókörünkön teljesen kívül esik, hiszen még megtalálásukra sincs idő. Ha egy mindentudó lény megadná nekünk az egyszerű láncokat, előfordulhatna, hogy azokat érthetetlennek tartanánk, mivel lehetséges, hogy a lánc jóságának legrövidebb bizonyítása hosszabb lenne, mint agysejtjeink száma.

Az S tétel elutasítja az ilyen forgatókönyvet a legtöbb Boole-függvény esetén. De kevesebb mint 2^{100} Boole-függvénynek lehet még a világtörténelem során gyakorlati jelentősége és az S tétel ezekről semmit sem mond.

1974-ben Stockmeyer és Meyer konstruáltak egy olyan f Boole-függvényt [56], aminek bizonyíthatóan nagyon nagy a bonyolultsága. Az ő f-jük nem „explicit", a fent leírt értelemben, de egyáltalán nem művi; a matematikai logikában merült fel. Tekintsük a következő szimbolikus állításokat:

$$\mathtt{048+1015\neq 1063;} \tag{62}$$
$$\forall \mathtt{m} \exists \mathtt{n}(\mathtt{m}<\mathtt{n}+1); \tag{63}$$
$$\forall \mathtt{n} \exists \mathtt{m}(\mathtt{m}+1<\mathtt{n}); \tag{64}$$
$$\forall \mathtt{a} \forall \mathtt{b}(\mathtt{b}\geq \mathtt{a}+2\Rightarrow \exists \mathtt{ab}(\mathtt{a}<\mathtt{ab}\wedge \mathtt{ab}<\mathtt{b})); \tag{65}$$
$$\forall \mathtt{A} \forall \mathtt{B}(\mathtt{A}\equiv \mathtt{B}\Leftrightarrow \neg\exists \mathtt{n}(\mathtt{n}\in \mathtt{A}\wedge \mathtt{n}\notin \mathtt{B}\vee \mathtt{n}\in \mathtt{B}\wedge \mathtt{n}\notin \mathtt{A})); \tag{66}$$
$$\forall \mathtt{A}(\exists \mathtt{n}(\mathtt{n}\in \mathtt{A}) \Rightarrow \exists \mathtt{m}(\mathtt{m}\in \mathtt{A}\wedge\forall \mathtt{n}(\mathtt{n}\in \mathtt{A}\Rightarrow \mathtt{m}\leq \mathtt{n}))); \tag{67}$$
$$\forall \mathtt{A}(\exists \mathtt{n}(\mathtt{n}\in \mathtt{A}) \Rightarrow \exists \mathtt{m}(\mathtt{m}\in \mathtt{A}\wedge\forall \mathtt{n}(\mathtt{n}\in \mathtt{A}\Rightarrow \mathtt{m}\geq \mathtt{n}))); \tag{68}$$
$$\exists \mathtt{P}\forall \mathtt{a}((\mathtt{a}\in \mathtt{P}\Leftrightarrow \mathtt{a}+3\notin \mathtt{P}) \Leftrightarrow \mathtt{a}<1000); \tag{69}$$
$$\forall \mathtt{A}\forall \mathtt{B}(\forall \mathtt{C}\forall \mathtt{c}(\mathtt{C}\equiv \mathtt{A}\wedge \mathtt{c}=1\vee \mathtt{c}\equiv \mathtt{B}\wedge \mathtt{c}=0\Rightarrow (\forall \mathtt{n}(\mathtt{n}\in \mathtt{C}\Leftrightarrow \mathtt{n}+1\in \mathtt{C}) \Leftrightarrow \mathtt{c}=1)) \Rightarrow \neg \mathtt{A}\equiv \mathtt{B}). \tag{70}$$

Stockmeyer és Meyer definiáltak egy L nyelvet, aminek az ábécéje a következő 63 karakterből áll:

$$\forall\exists\neg()\equiv\in\notin+\wedge\vee\Rightarrow\Leftrightarrow<\leq=\neq\geq>\mathtt{abcdefghijklmnopqABCDEFGHIJKLMNOPQ0123456789}$$

és a szimbólumoknak megfelelő értelmezést adtak. Kisbetűk egy sorozata az L egy mondatán belül – a (65)-ban lévő 'ab'-hez hasonlóan – numerikus változókat jelentenek, amelyek a nemnegatív egész számokat futják be; a nagybetűk sorozatai halmazváltozókat jelentenek, amelyek egész számok véges részhalmazait futják be. Például a (66) jelentése: „Minden véges A és B halmazra, $A = B$ akkor és csak akkor, ha nincs olyan n szám, ami benne van A-ban, de nincs benne B-ben, vagy benne van B-ben de nincs benne A-ban." Ezen állítások némelyike igaz, mások hamisak. (Lásd a 82. gyakorlatot.)

A (62)–(70)-ben lévő sorozatok az L-hez tartoznak, de ez a nyelv nagyon korlátos. A számokon megengedett egyetlen aritmetikai művelet egyetlen konstans hozzáadása. Azt írhatjuk, hogy 'a+13', de azt nem,hogy 'a+b'. A megengedett egyetlen reláció egy szám és egy halmaz között az eleme reláció ($\in$ vagy $\notin$). A halmazok között megengedett egyetlen reláció az egyenlőség ($\equiv$). Ezenkívül minden változót kvantálni kell a $\forall$ vagy a $\exists$ kvantorral.[2]

L-nek minden $k \leq n$ hosszúságú mondata reprezentálható egy $6n$ hosszúságú bináris vektorral, ahol az utolsó $6(n-k)$ biten nullák vannak. Legyen $f(x)$ egy olyan $6n$ változós Boole-függvény, ahol $f(x) = 1$, ha x az L egy igaz mondata, és $f(x) = 0$, ha x egy hamis mondatot reprezentál. Az $f(x)$ értéke nem specifikált, ha x egy nem értelmes mondatot reprezentál. Egy ilyen f függvény igazságtáblája véges számú lépésben előállítható Büchi és Elgot tételeinek megfelelően [7, 16].

De a „véges" nem jelenti azt, hogy „megcsinálható": Stockmeyer és Meyer bizonyította, hogy

$$C(f) > 2^{r-5}, \qquad \text{ha } n \geq 460 + .302r + 5.08 \ln r \text{ és } r > 36. \tag{71}$$

Végeredményben, ha $n = 621$, akkor $C(f) > 2^{426} > 10^{128}$. *Ennyi sok kapuból álló Boole-lánc sohasem építhető meg*, mivel 10^{128} egy nagyvonalú felső becslés az univerzumbeli összes protonok számára. Ez egy meglehetősen kicsi, véges probléma, ami megoldva soha nem lesz. Stockmeyer és Meyer bizonyításának részletei a [58] cikkben jelentek meg. Az

[2]A szó szoros értelmében az L mondatai„az egyetlen rákövetkezés függvénnyel rendelkező gyenge másodrendű monadikus logikához" tartoznak. A gyenge másodrendű logika megengedi a véges halmazok feletti kvantifikálást. A k rákövetkezés függvénnyel rendelkező monadikus logika a címkézetlen k-fák elmélete.

alapötlet az, hogy bár az L nyelv korlátozott, de elég gazdag igazságtáblák és Boole-láncok összetettségének eléggé rövid mondatokkal való leírásához. Így aztán f-nek olyan inputokat kell kezelni, amelyek önmagukra hivatkoznak.

További olvasnivaló. Jelentős cikkek ezreit írták a Boole-kapukból álló hálózatokról, mivel az ilyen hálózatok az elmélet és a gyakorlat sok vonatkozásában alapvetőek. Ebben a részben olyan témakörökre koncentrálunk, amelyek a szekvenciális számítógépek programozásával vannak kapcsolatban. Azonban a párhuzamos számításokkal kapcsolatos témákat is behatóan vizsgáljuk, mint például a kis mélységű hálózatok, amelyekben a kapuk bemenetszáma tetszőleges („nem korlátozott bemenet"). Wegener a [63] könyvében a teljes tárgykör nagyon jó bevezetését nyújtja.

Főleg olyan Boole-láncokkal foglalkozunk, amelyekben minden bináris művelet egyformán fontos. A mi szempontunkból, az olyan kapuk mint a $\oplus$ vagy a $\overline{\subset}$ nem kevésbé fontosak, mint az $\wedge$ vagy a $\vee$ kapuk. De az is természetes, hogy ha monoton függvényt akarunk kiszámítani, úgy gondoljuk, hogy elegendő csak monoton műveleteket használni, mint a $\wedge$ és a $\vee$. Razborov meglepő bizonyítási technikát fejlesztett ki annak belátására, hogy a monoton műveletek önmagukban korlátozott képességűek. Megmutatta például, hogy minden olyan ÉS-VAGY lánc költsége biztosan $n^{\Omega(\lg n)}$, amellyel egy 0 és 1-esekből álló $n \times n$-es mátrix permanenséről meg akarjuk állapítani, hogy nulla vagy egy lásd [41, 42]. Ezzel szemben a 7.5.1. pontban látni fogjuk, hogy ez a probléma ekvivalens a „páros illesztés"-sel, ami mindössze $O(n^{2.5})$ lépésben megoldható. Továbbá ebben a pontban a hatékony módszereket kissé nagyobb költségű Boole-láncokként lehet implementálni, ahol megengedjük az $\wedge$-en és a $\vee$-on kívül a negáció vagy más Boole-műveletek használatát is. (Pratt ezt „a negatív gondolkodás erejének" nevezte.) Razborov módszeréről egy bevezetés található a 85. és a 86. gyakorlatban.

GYAKORLATOK

1. [*24*] A (6)-os formula által leírt „véletlen" függvény egy 4 költségű, 4 mélységű Boole-láncnak felel meg. Keressünk olyan 3 mélységű formulát, melynek költsége ugyanannyi.

2. [*21*] Mutassunk olyan 3 mélységű, 5 költségű formulát, mellyel kiszámítható (a) $w \oplus \langle xyz \rangle$, illetve (b) $w \wedge \langle xyz \rangle$.

3. [*M23*] [18] Bizonyítsuk be, hogy ha egy $f(x_1, \ldots, x_n)$ Boole-függvény pontosan k pontban igaz, akkor $L(f) < 2n + (k-2)\,2^{k-1}$. Tekintsük a $k = 3$, $n = 10^6$ esetet.

4. [*M26*] [54] Igazoljuk, hogy bármely f Boole-függvény esetén a minimális mélységre, illetve formula hosszra teljesül, hogy $\lg L(f) < D(f) \le \alpha \lg L(f) + 1$, ahol $\alpha = 2/\lg(\frac{3}{2}) \approx 3.419$. *Útmutatás.* Minden $r \ge 3$ belső csúcsot tartalmazó bináris fában létezik s belső csúcsból álló részfa, melyre $\frac{1}{3}r \le s < \frac{2}{3}r$.

5. [*21*] Az $F_n(x_1, \ldots, x_n) = \left\langle x_1^{F_1} x_2^{F_2} \ldots x_{n-1}^{F_{n-1}} x_n^{F_{n-2}} \right\rangle$ Fibonacci küszöbfüggvényt, ahol $n \ge 3$, már elemeztük a **??** gyakorlatban. Létezik-e hatékony módszer a kiértékelésére?

6. [*20*] Igaz vagy hamis: egy $f(x_1, \ldots, x_n)$ függvény akkor és csak akkor normalizált, ha eleget tesz az $f(x_1, \ldots, x_n) \wedge y = f(x_1 \wedge y, \ldots, x_n \wedge y)$ általános disztributív szabálynak.

7. [*20*] Alakítsuk át az '$x_5 = x_1 \overline{\vee} x_4$, $x_6 = \bar{x}_2 \vee x_5$, $x_7 = \bar{x}_1 \wedge \bar{x}_3$' Boole-láncot egy vele ekvivalens $(\hat{x}_5, \hat{x}_6, \hat{x}_7, \hat{x}_8)$ lánccá, melyben minden lépés normalizált.

8. [*20*] Magyarázzuk meg, hogy (11) miért lehet az x_k változó igazságtáblája.

9. [*20*] Az L algoritmus tetszőleges f függvényhez megadja a legrövidebb formula hosszát, de semmilyen további információval nem szolgál. Egészítsük ki az algoritmust úgy, hogy az egy, (6)-hoz hasonló tényleges minimális hosszúságú formulát is megadjon.

10. [*20*] Módosítsuk az L algoritmust oly módon, hogy az $L(f)$ helyett $D(f)$-et számítsa ki.

11. [*22*] Módosítsuk az L algoritmust úgy, hogy az $L(f)$ helyett az $U(f)$ felső korlátot, valamint a szövegben bemutatott $\phi(f)$ lábnyomot számítsa ki.

12. [*15*] Milyen Boole-lánc felel meg a (13)-as minimális-memóriájú sémának?

13. [*16*] Milyen igazságtáblák tartoznak a (13)-as példa f_1, f_2, f_3, f_4 és f_5 függvényeihez?

14. [*22*] Hogyan lehet alkalmas módon meghatározni a (17)-es függvénnyel leírt $5n(n-1)$ igazságtáblát, ha ismerjük g igazságtábláját? (Használjunk bitenkénti műveleteket, mint (15)-ben és (16)-ban.)

15. [*28*] Hogyan számíthatóak ki a lehető legrövidebb módon az alábbi függvények minimális memória felhasználás mellett: (a) $S_2(x_1, x_2, x_3, x_4)$; (b) $S_1(x_1, x_2, x_3, x_4)$; (c) a (18) függvény.

16. [*F33*] Lássuk be, hogy a 2^{128} $f(x_1, \ldots, x_7)$ Boole-függvényből kevesebb, mint 2^{118} számítható ki minimális memóriával.

17. [*25*] (M. S. Paterson, 1977.) Bár az $f(x_1, \ldots, x_n)$ Boole-függvényt nem lehet mindig n regiszterben kiértékelni, lássuk be, hogy $n+1$ regiszter viszont mindig elegendő. Más szóval, mutassuk meg, hogy ha feltesszük, hogy $0 \le j(i), k(i) \le n$, akkor mindig van a (13)-hoz hasonló műveletsorozat az $f(x_1, \ldots, x_n)$ kiszámítására.

18. [*35*] Vizsgáljuk meg az $f(x_1, x_2, x_3, x_4, x_5)$ kiszámítását optimális minimális-memóriával: hány olyan ötváltozós függvényosztály létezik, ahol $C_m(f) = r$ (ha $r = 0, 1, 2, \ldots$)?

19. [*M22*] Igazoljuk, hogy ha egy Boole-láncban n változót használunk fel és a lánc hossza $r < n+2$, akkor igazoljuk, hogy létezik vagy egy fentről-lefelé vagy egy lentről-felfelé előállítás.

20. [*40*] (R. Schroeppel, 2004) Egy Boole-lánc „kényszerítő”, ha az $\oplus$ vagy $\equiv$ műveleti jelek egyikét sem használja. Keressük meg az összes 4 változós függvényre, a fenti megszorítások mellett az optimális költséget, hosszt és mélységet. Ad-e a lábnyom heurisztika optimális eredményeket?

21. [*46*] Hány darab négyváltozós függvényhez találtak a Harvard kutatói optimális elektroncsöves áramkört 1951-ben?

22. [*21*] Mutassuk meg, hogy a 7.10. ábrán az S_3-hoz tartozó lánc tulajdonképpen magában foglalja a 9. ábrán lévő $S_{2,3}$-hoz tartozó láncot. Keressünk hasonló láncot $S_2(x_1, x_2, x_3, x_4, x_5)$-re.

23. [*23*] A 7.10. ábra a lehetséges 64 ötváltozós szimmetrikus függvény közül csak 16-ot illusztrál. Mutassuk meg, hogyan írjuk fel az optimális láncokat a többiekre.

24. [*47*] Igaz-e, hogy minden szimmetrikus f függvényre $C_m(f) = C(f)$?

25. [*17*] Tegyük fel, hogy olyan Boole-láncot akarunk, ami magában foglal *minden* n változós Boole-függvényt. Legyen $f_k(x_1, \ldots, x_n)$ egy olyan függvény, ahol k bináris ábrázolása a függvény igazságtáblája, $0 \le k < m = 2^{2^n}$-re. Mi a $C(f_0 f_1 \ldots f_{m-1})$?

26. [*25*] Igaz vagy hamis: Ha $f(x_0, \ldots, x_n) = (x_0 \wedge g(x_1, \ldots, x_n)) \oplus h(x_1, \ldots, x_n)$, ahol g és h nemtriviális Boole-függvények és együttes költségük $C(gh)$, akkor $C(f) = 2 + C(gh)$.

27. [*23*] Lehet-e a (22) teljes összeadót mindössze öt lépéssel implementálni minimális memória felhasználása mellett (azaz mindössze három egy bites regiszterrel)?

28. [*26*] Bizonyítsuk be, hogy $C(u'v') = C(u''v'') = 5$ teljesül a következő kétkimenetű függvényekre:

$$(u'v')_2 = \big(x + y - (uv)_2\big) \mod 4, \qquad (u''v'')_2 = \big(-x - y - (uv)_2\big) \mod 4.$$

Használjuk ezeket a függvényeket $[(x_1 + \cdots + x_n) \mod 4 = 0]$ kevesebb, mint $2.5n$ lépésben való kiértékelésére.

29. [*M28*] Bizonyítsuk be, hogy a (21) képlettel kapcsolatban definiált *teljes összeadó áramkör* a (27) *általános oldalösszeg* kiszámítására bemutatott áramkör mélysége $O(\lg n)$.

30. [*M25*] Oldjuk meg az oldalösszeadások $s(n)$ költségére vonatkozó (28) bináris rekurzív egyenletet. az oldalsó összegek

31. [*21*] Bizonyítsuk be, hogy $C(f) \le 5n + O(n/\log n)$, ha $f(x_1, \ldots, x_n)$ szimmetrikus.

32. [*F16*] Miért teljesül (30) megoldására $t(n) = 2^n + O(2^{n/2})$?

33. [*F22*] Igaz vagy hamis: Ha $1 \le N \le 2^n$ és $n \to \infty$ és $N \to \infty$, akkor $\{x_1, \ldots, x_n\}$ első N mintermje kiértékelhető $N + O(\sqrt{N}\,)$ lépésben.

34. [*22*] Egy *prioritás kódolónak* $n = 2^m - 1$ bemenete van – jelölje $x_1 \ldots x_n$ – és m kimenete – $y_1 \ldots y_m$ –, ahol $(y_1 \ldots y_m)_2 = k$ pontosan akkor igaz, ha $k = \max\{j \mid j = 0 \text{ vagy } x_j = 1\}$. Tervezzünk egy ilyen kódolót $O(n)$ költséggel és $O(m)$ mélységgel.

35. [*23*] Mutassuk meg, hogy ha $n > 1$, akkor az $x_1 \wedge \cdots \wedge x_{k-1} \wedge x_{k+1} \wedge \cdots \wedge x_n$ $(1 \le k \le n)$ konjunkciók kiszámíthatóak $(x_1, \ldots, x_n)$-ből legfeljebb $3n - 6$ teljes költséggel.

36. [*M28*] [29] Legyen y_k az $x_1 \wedge \cdots \wedge x_k$ „prefix” minden $1 \le k \le n$-re. Nyilvánvaló, hogy $C(y_1 \ldots y_n) = n - 1$ és $D(y_1 \ldots y_n) = \lceil \lg n \rceil$, de nem tudjuk egyszerre minimalizálni a költséget és a mélységet. Keressünk egy olyan optimális mélységű láncot, melynek a költsége $< 4n$.

37. [*M28*] (Marc Snir, 1986.) [53] Adott $n \ge m \ge 1$ esetén tekintsük a következő algoritmust:

S1. (Előre haladó ciklus.) Minden $t \leftarrow 1, 2, \ldots, \lceil \lg m \rceil$-re legyen $x_{\min(m, 2^t k)} \leftarrow x_{2^t(k-1/2)} \wedge x_{\min(m, 2^t k)}$ minden $k \ge 1$ és $2^t(k - 1/2) < m$ esetén.
S2. (Visszafelé haladó ciklus.) Minden $t \leftarrow \lceil \lg m \rceil - 1, \lceil \lg m \rceil - 2, \ldots, 1$-re legyen $x_{2^t(k+1/2)} \leftarrow x_{2^t k} \wedge x_{2^t(k+1/2)}$ minden $k \ge 1$ és $2^t(k + 1/2) < m$ esetén.
S3. (Kiterjesztés.) Minden $k \leftarrow m + 1, m + 2, \ldots, n$-re legyen $x_k \leftarrow x_{k-1} \wedge x_k$. ■

a) Bizonyítsuk be, hogy az algoritmus megoldja a 36. gyakorlat prefix problémáját, azaz $(x_1, x_2, \ldots, x_n)$-et $(x_1, x_1 \wedge x_2, \ldots, x_1 \wedge x_2 \wedge \cdots \wedge x_n)$-re képezi le.

b) Legyen $c(m, n)$ és $d(m, n)$ a költsége és a mélysége a megfelelő Boole-láncnak. Bizonyítsuk be, hogy $c(m, n) + d(m, n) = 2n - 2$, ha n elég nagy.

c) Mit jelent adott n esetén a $d(n) = \min_{1 \le m \le n} d(m, n)$ érték? Mutassuk meg, hogy $d(n) < 2 \lg n$.

d) Bizonyítsuk be, hogy létezik $2n - 2 - d$ költségű és d mélységű Boole-lánc a prefix probléma megoldására, ha $d(n) \le d < n$. (A 81. gyakorlat alapján ez a költség optimális is.)

38. [*25*] Az 5.3.4. pontban megvizsgáltuk a *rendező hálózatokat* és azt kaptuk, hogy $\hat{S}(n)$ összehasonlító egység elegendő n szám – $(x_1, x_2, \ldots, x_n)$ – növekvő sorrendbe rendezéséhez. Ha az x_j bemenetek csak 0-k és 1-ek lehetnek, minden összehasonlító ekvivalens két kapuval, $(x \wedge y, x \vee y)$-nal. Így egy rendező hálózat megfeleltethető egy Boole-láncnak, ami $(x_1, x_2, \ldots, x_n)$-nek n részfüggvényét számítja ki.

a) Melyik az az n függvény – $f_1 f_2 \ldots f_n$ –, amelyeket egy rendező hálózat kiszámít?

b) Mutassuk meg, hogy az $\{f_1, f_2, \ldots, f_n\}$ függvények kiszámíthatóak $O(n)$ lépésben egy $O(\lg n)$ mélységű lánccal. (Tehát a rendező hálózatok a Boole-függvényekre alkalmazva aszimptotikusan nem optimálisak.)

39. [*M21*] (Paterson és Klein [38].) Implementáljuk (31) $M_m(x_1, \ldots, x_m; y_0, y_1, \ldots, y_{2^m-1})$ 2^m-es multiplexerét egy hatékony lánccal, ami egyszersmind igazolja a $C(M_m) \le 2n + O(\sqrt{n})$ és $D(M_m) \le m + O(\lg m)$ felső korlátokat is.

40. [*25*] Ha $n \geq k \geq 1$, legyen $f_{nk}(x_1, \ldots, x_n)$ a „egy sorban k" függvény, azaz

$$(x_1 \wedge \cdots \wedge x_k) \vee (x_2 \wedge \cdots \wedge x_{k+1}) \vee \cdots \vee (x_{n+1-k} \wedge \cdots \wedge x_n).$$

Mutassuk meg, hogy a $C(f_{nk})$ költség kevesebb, mint $4n - 3k$.

41. [*M23*] (*Feltételes összeg összeadók.*) A (25) bináris összeadás $O(\lg n)$ mélység melletti megvalósításának egyik módja a 4. gyakorlat multiplexeres trükkjén alapszik: Ha $(xx')_2+(yy')_2 = (zz')_2$, ahol $|x'| = |y'| = |z'|$, akkor vagy $(x)_2 + (y)_2 = (z)_2$ és $(x')_2 + (y')_2 = (z')_2$ teljesül, vagy pedig $(x)_2 + (y)_2 + 1 = (z)_2$ és $(x')_2 + (y')_2 = (1z')_2$. Hogy időt takarítsunk meg, kiszámíthatjuk $(x)_2 + (y)_2$ és $(x)_2 + (y)_2 + 1$ értékét egyszerre mialatt $(x')_2 + (y')_2$-t számítjuk ki. Utána, ha már tudjuk, hogy $(x')_2 + (y')_2$ kevésbé szignifikáns része eredményez-e átvitelt, multiplexerekkel kiválaszthatjuk a helyes biteket a legszignifikánsabb részhez. Ha $n = 2^m$ és ezt módszert rekurzívan alkalmazzuk $2n$-bites összeadók építésére n-bites összeadókból, hány kapura van szükség? Mekkora lesz a mélység?

42. [*25*] A (25) bináris összeadásban legyen $u_k = x_k \wedge y_k$ és $v_k = x_k \oplus y_k$ minden $0 \leq k < n$ esetén.

a) Mutassuk meg, hogy $z_k = v_k \oplus c_k$, ahol a c_k átvitel bitekre teljesül, hogy

$$c_k = u_{k-1} \vee (v_{k-1} \wedge (u_{k-2} \vee (v_{k-2} \wedge (\cdots (u_1 \wedge v_0) \cdots)))) .$$

b) Legyen $U_k^k = 0$, $V_k^k = 1$ és $U_j^{k+1} = u_k \vee (v_k \wedge U_j^k)$, $V_j^{k+1} = v_k \wedge V_j^k$, ha $k \geq j$. Bizonyítsuk be, hogy $c_k = U_0^k$ és $U_i^k = U_j^k \vee (V_j^k \wedge U_i^j)$, $V_i^k = V_j^k \wedge V_i^j$, ha $i \leq j \leq k$.

c) Legyen $h(m) = 2^{m(m-1)/2}$. Mutassuk meg, ha $n = h(m)$, akkor a $c_1, \ldots, c_n$ átvitel bitek mind kiértékelhetőek úgy, hogy a mélység $(m+1)m/2 \approx \lg n + \sqrt{2 \lg n}$ és a teljes költség $O(2^m n)$.

43. [*28*] Egy ***véges állapotú átalakító*** egy absztrakt gép A véges bemenet ábécével, B véges kimenet ábécével és a belső állapotok véges Q halmazával. Az állapotok egyike, q_0, az úgynevezett „kezdő állapot". Egy adott $\alpha = a_1 \ldots a_n$ füzérre, ahol minden $a_j \in A$, a gép kiszámítja a $\beta = b_1 \ldots b_n$ füzért, ahol minden $b_j \in B$, a következőféleképpen:

T1. (Kezdeti értékek beállítása.) Legyen $j \leftarrow 1$ és $q \leftarrow q_0$.
T2. (Kész?) Az algoritmus befejeződik, ha $j > n$.
T3. (b_j kimenet.) Legyen $b_j \leftarrow c(q, a_j)$.
T4. (j növelése.) Legyen $q \leftarrow d(q, a_j)$, $j \leftarrow j + 1$, és folytassuk a T2 lépéssel. ■

A gép beépített utasításokkal rendelkezik a $c(q, a) \in B$ és $d(q, a) \in Q$ értékek meghatározására minden $q \in Q$ állapot és minden $a \in A$ karakter esetében. A gyakorlat célja, hogy megmutassuk, hogy ha egy ilyen véges állapotú átalakító A és B ábécéit binárisan kódoljuk, akkor a β füzér kiszámítható α-ból egy $O(n)$ méretű és $O(\lg n)$ mélységű Boole-lánccal.

a) Tekintsük azt a problémát, hogyan lehet egy $a_1 \ldots a_n$ bináris vektort $b_1 \ldots b_n$-re változtatni, ahol

$$b_j \;\leftarrow\; a_j \oplus [a_j = a_{j-1} = \cdots = a_{j-k} = 1 \text{ és } a_{j-k-1} = 0, \text{ ahol } k \text{ páratlan}],$$

feltéve, hogy $a_0 = 0$. Például $\alpha = 1100100100011111101101010 \mapsto \beta = 1000100100010101001001010$. Bizonyítsuk be, hogy ezt a transzformációt meg lehet valósítani egy véges állapotú átalakítóval úgy, hogy $|A| = |B| = |Q| = 2$.

b) Tegyük fel, hogy egy véges állapotú átalakító a q_j állapotba került $a_1 \ldots a_{j-1}$ beolvasása után. Hogyan lehet a $q_1 \ldots q_n$ sorozatot kiszámítani $O(n)$ költségű és $O(\lg n)$ mélységű Boole-lánccal, felhasználva Ladner és Fischer konstrukcióját a 36. gyakorlatból? (Ebből a $q_1 \ldots q_n$ sorozatból már könnyen kiszámítható $b_1 \ldots b_n$, hiszen $b_j = c(q_j, a_j)$.)

c) Alkalmazzuk a (b)-ben látott módszert az (a)-ban bemutatott problémára.

44. [*26*] [29] Mutassuk meg, hogy a (25) bináris összeadás problémájára lehet úgy is tekinteni, mint egy véges állapotú átalakításra. Jellemezzük a 43. gyakorlat konstrukciójával kapható Boole-láncot, ha $n = 2^m$, és hasonlítsuk össze a 41. gyakorlat feltételes összeg összeadójával.

45. [*F20*] Miért nem érvelhetünk az S tétel bizonyításában egyszerűen azzal, hogy összesen $n^2(n+1)^2 \ldots (n+r-1)^2$-féleképpen lehet megválasztani $j(i)$ és $k(i)$ értékét úgy, hogy $1 \le j(i), k(i) < i$?

46. [*F21*] Legyen $\alpha(n) = c\bigl(n, \lfloor 2^n/n \rfloor\bigr)/2^{2^n}$ azon $f(x_1, \ldots, x_n)$ n változós Boole-függvények aránya, amelyekre $C(f) \le 2^n/n$. Bizonyítsuk be, hogy $\alpha(n)$ gyorsan tart nullához, ha $n \to \infty$.

47. [*M23*] Terjesszük ki az S tételt az n bemenetű m kimenetű függvényekre is.

48. [*F23*] Határozza meg a legkisebb olyan $r = r(n)$ egész értéket, amelyre $(r-1)!2^{2^n} \le 2^{2r+1}(n+r-1)^{2r}$ teljesül, (a) pontosan, ha $1 \le n \le 16$; (b) aszimptotikusan, ha $n \to \infty$.

49. [*F25*] Bizonyítsuk be, hogy ha $n \to \infty$, akkor majdnem minden $f(x_1, \ldots, x_n)$ Boole-függvény minimális hosszára $L(f) > 2^n/\lg n - 2^{n+2}/(\lg n)^2$ teljesül.

50. [*24*] Melyek a (35) prímszám-függvény prím implikánsai és prím klózai? Írjuk fel a függvényt minimális hosszú (a) DNF (b) KNF alakban.

51. [*20*] A prímszám felismerő mely reprezentációja helyettesíti (37)-et, ha az igazságtábla sorait x_5x_6 alapján készítjük el x_1x_2 helyett?

52. [*23*] Mely k és l értékek mellett lesz a (38) felső korlát minimális, ha $5 \le n \le 16$?

53. [*F22*] Becsüljük meg (38) értékét, ha $k = \lfloor 2 \lg n \rfloor$, $l = \lceil 2^k/(n - 3\lg n) \rceil$ és $n \to \infty$.

54. [*29*] Keressünk egy rövid Boole-láncot a következő hat függvény kiértékelésére: $f_j(x) = [x_1x_2x_3x_4 \in A_j]$, ahol $A_1 = \{0010, 0101, 1011\}$, $A_2 = \{0001, 1111\}$, $A_3 = \{0011, 0111, 1101\}$, $A_4 = \{1001, 1111\}$, $A_5 = \{1101\}$, $A_6 = \{0101, 1011\}$. (Ez a hat függvény szerepel a (37) prímszám felismerőben.) A kapott láncot hasonlítsuk össze Lupanov általános módszerének minterm-először kiértékelési sémájával.

55. [*34*] Mutassuk meg, hogy a 6 bites prímszám felismerő függvény költsége legfeljebb 14.

56. [*16*] Magyarázzuk meg, hogy miért nulla költségű valamennyi 14 vagy több közömbös hellyel rendelkező függvény a 7.8. táblázatban.

57. [*19*] Mely hét szegmenses „számjegyek” kerülnek megjelenítésre, ha $(x_1x_2x_3x_4)_2 > 9$ (45)-ben?

58. [*30*] Egy 4×4 bites *S-doboz* a $\{0000, 0001, \ldots, 1111\}$ 4 bites vektorok egy permutációja. Az ilyen permutációkat ismert kriptográfiai rendszerben szokták felhasználni, mint amilyen az orosz GOST 28147 (1989) szabvány. Minden 4×4 bites S-doboz megfelel négy függvény, $f_1(x_1, x_2, x_3, x_4)$, …, $f_4(x_1, x_2, x_3, x_4)$ sorozatának, ezáltal egy $x_1x_2x_3x_4 \mapsto f_1f_2f_3f_4$ leképezést megvalósítva.

Találjuk meg az összes olyan 4×4-bites S-dobozt, amelyre $C(f_1) = C(f_2) = C(f_3) = C(f_4) = 7$.

59. [*29*] Az 58. gyakorlat feltételeit kielégítő egyik S-doboz $(\mathtt{0}, \ldots, \mathtt{f}) \mapsto (\mathtt{0,6,5,b,3,9,f,e,c,4,7,8,d,2,a,1})$, másképpen fogalmazva (f_1, f_2, f_3, f_4) igazságtáblái rendre $(\mathtt{179a}, \mathtt{63e8}, \mathtt{5b26}, \mathtt{3e29})$. Keressünk egy Boole-láncot, ami ezt a négy „maximálisan bonyolult” függvényt kevesebb, mint 20 lépésben kiértékeli.

60. [*23*] (Frank Ruskey.) Tegyük fel, hogy $z = (x + y) \mod 3$, ahol $x = (x_1x_2)_2$, $y = (y_1y_2)_2$, $z = (z_1z_2)_2$, és minden kétbites érték csak 00, 01 vagy 10 lehet. Számítsuk ki z_1 és z_2 értékét x_1, x_2, y_1, y_2 értékéből hat Boole-lépésben.

61. [*34*] A 60. gyakorlatot folytatva keressünk egy jó módszert $z = (x+y) \mod 5$ kiszámítására, ezúttal a következő három bites értékeket megengedve: 000, 001, 010, 011, 100.

62. [*F23*] Tekintsünk egy olyan véletlen parciális n változós Boole-függvényt, amelynek a 2^nc

helyen adott az értéke és $2^n d$ helyen közömbös, ahol $c + d = 1$. Bizonyítsuk be, hogy majdnem minden ilyen parciális függvény költsége több, mint $2^n c/n$.

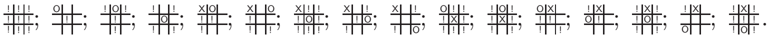
63. [*F35*] (L. A. Sholomov, 1969 [51].) A 62. gyakorlatot folytatva bizonyítsuk be, hogy valamennyi fenti tulajdonságú függvény költsége $\le 2^n c/n(1 + O(n^{-1} \lg n))$. *Útmutatás:* Létezik egy olyan $2^m(1 + k)$ elemű $x_1 \ldots x_k$ vektorokból álló halmaz, amely belemetsz a k-dimenziós kocka minden $(k - m)$-dimenziós részkockájába.

64. [*25*] (*Bűvös tizenöt.*) Két játékos felváltva választ számjegyeket 1-től 9-ig, minden számjegyet legfeljebb egyszer felhasználva. Ha van egyáltalán, akkor a győztes az, aki először tesz szert három olyan számjegyre, hogy azok összege éppen 15. Milyen stratégiát lenne érdemes használni?

65. [*35*] Módosítsuk a (47)–(56)-ban bemutatott amőba stratégiát úgy, hogy minden esetben jó döntést hozzon.

66. [*20*] Értékeljük a 65. gyakorlatban választott lépéseket. Mindig optimálisak?

67. [*40*] Ahelyett, hogy csak egyetlen jó lépést keresnénk minden egyes pozícióra a amőba játékban, kereshetnénk az összeset is. Máshogy fogalmazva adott $x_1 \ldots x_9 o_1 \ldots o_9$-hez megpróbálhatnánk kiszámítani a kilenc elemű $g_1 \ldots g_9$ kimenetet, ahol $g_j = 1$ akkor és csak akkor, ha a j mező választása X számára jó lépés lenne. Például a következő tipikus esetekben ezek adódnak X felkiáltójellel jelölt jó lépéseire:



Olyan gép ellen játszani sokkal nagyobb élvezet, amely ezek közül a lehetőségek közül választ véletlenszerűen, mint egy olyan ellen, ami mindig ugyanazt lépi. Az összes-jó-lépés probléma egyik vonzó megoldása annak kihasználása, hogy az amőbának nyolc szimmetriája van. Képzeljünk el egy áramkört, amelynek 18 bemenete – $x_1 \ldots x_9 o_1 \ldots o_9$ – és három kimenete – (c, s, m) – van, melyek jelentése rendre „sarok," „oldal" és „közép". Továbbá legyen elegendő nyolc ilyen áramkört megfelelő módon összekapcsolni az előállítani kívánt g_j függvények kiszámításához:

$$\begin{aligned}
g_1 &= c(x_1x_2x_3x_4x_5x_6x_7x_8x_9o_1o_2o_3o_4o_5o_6o_7o_8o_9)\\
&\qquad \vee c(x_1x_4x_7x_2x_5x_8x_3x_6x_9o_1o_4o_7o_2o_5o_8o_3o_6o_9),\\
g_2 &= s(x_1x_2x_3x_4x_5x_6x_7x_8x_9o_1o_2o_3o_4o_5o_6o_7o_8o_9)\\
&\qquad \vee s(x_3x_2x_1x_6x_5x_4x_9x_8x_7o_3o_2o_1o_6o_5o_4o_9o_8o_7),\\
g_3 &= c(x_3x_2x_1x_6x_5x_4x_9x_8x_7o_3o_2o_1o_6o_5o_4o_9o_8o_7)\\
&\qquad \vee c(x_3x_6x_9x_2x_5x_8x_1x_4x_7o_3o_6o_9o_2o_5o_8o_1o_4o_7),\\
g_4 &= s(x_1x_4x_7x_2x_5x_8x_3x_6x_9o_1o_4o_7o_2o_5o_8o_3o_6o_9)\\
&\qquad \vee s(x_7x_4x_1x_8x_5x_2x_9x_6x_3o_7o_4o_1o_8o_5o_2o_9o_6o_3), \qquad \ldots\\
g_9 &= c(x_9x_8x_7x_6x_5x_4x_3x_2x_1o_9o_8o_7o_6o_5o_4o_3o_2o_1)\\
&\qquad \vee c(x_9x_6x_3x_8x_5x_2x_7x_4x_1o_9o_6o_3o_8o_5o_2o_7o_4o_1),
\end{aligned}$$

és g_5 mind a nyolc áramkör m kimenetének összevagyolásával adódik.

Tervezzünk egy ilyen áramkört kevesebb, mint 2000 kapu felhasználásával.

68. [*M25*] Tekintsük az n bites π-függvényt $\pi_n(x_1 \ldots x_n)$-et, melynek az értéke a π bináris reprezentációjának legszignifikánsabb bitjétől jobbra eső $(x_1 \ldots x_n)_2$-edik bit. Bizonyítja-e a 4.3.1-39. gyakorlatban π tetszőleges bitjeinek hatékony kiszámítására bemutatott módszer, hogy elég nagy n-re $C(\pi_n) < 2^n/n$?

69. [*M24*] Legyen f multilineáris reprezentációja

$$\alpha_{000} \oplus \alpha_{001}x_m \oplus \alpha_{010}x_j \oplus \alpha_{011}x_jx_m \oplus \alpha_{100}x_i \oplus \alpha_{101}x_ix_m \oplus \alpha_{110}x_ix_j \oplus \alpha_{111}x_ix_jx_m\,,$$

ahol minden α_l együttható az $\{x_1, \ldots, x_n\} \setminus \{x_i, x_j, x_m\}$ változók függvénye.

a) Bizonyítsuk be, hogy az (58) párok pontosan akkor „jók”, ha az együtthatók kielégítik a következő egyenleteket:

$$\alpha_{010}\alpha_{101} = \alpha_{011}\alpha_{100}, \quad \alpha_{101}\alpha_{110} = \alpha_{100}\alpha_{111} \quad \text{és} \quad \alpha_{110}\alpha_{011} = \alpha_{111}\alpha_{010}\,.$$

b) Mely (i, j, m) értékekre lesznek a párok rosszak, ha $f = (\det X) \mod 2$? Lásd (60).

70. [*M27*] Legyen X a 3×3-as Boole-mátrix (60)-ból. Keressünk hatékony láncokat a következő Boole-függvényekhez: (a) $(\det X) \mod 2$; (b) $[\operatorname{per} X > 0]$; (c) $[\det X > 0]$.

71. [*M26*] Tegyük fel, hogy $f(x)$ értéke 0 p valószínűséggel minden $x = x_1 \ldots x_n$ pontban, függetlenül a többi helyen felvett értékétől.

a) Milyen valószínűséggel lesznek az (58) párok jók?
b) Mi a valószínűsége annak, hogy léteznek rossz (58) párok?
c) Milyen valószínűséggel találunk rossz (58) párokat legfeljebb t véletlen próbával?
d) Várhatóan mennyi időbe telik az (i, j, m) eset tesztelése, mint p, t és n függvénye?

72. [*M24*] Terjesszük ki az előző feladatot parciális függvényekre, is, ahol $f(x) = 0$ p valószínűséggel, $f(x) = 1$ q valószínűséggel és $f(x) = *$ r valószínűséggel.

73. [*20*] Mutassuk meg, hogy ha minden (i, j, m), $m \neq i \neq j \neq m$, esetén létezik rossz (58) pár, akkor elegendő $\binom{n}{2}$ jól megválasztott (i, j, m) hármast megvizsgálni annak bizonyításához, hogy f nem dekomponálható.

74. [*25*] Terjesszük ki az előző gyakorlat ötletét és javasoljunk egy módszert az egymást követő (i, j, m) hármasok kiválasztására Shen, McKellar és Weiner módszerének alkalmazásához.

75. [*20*] Mi történik, ha a fejezet dekompozíciós eljárását az $S_{0,n}(x_1, \ldots, x_n)$ „mindegyik-azonos” függvényre alkalmazzuk?

76. [*M25*] (D. Uhlig, 1974.) [60] A gyakorlat célja, hogy bebizonyítsuk azt a meglepő tényt, hogy bizonyos f függvények esetében a legjobb lánc költsége az

$$F(u_1, \ldots, u_n, v_1, \ldots, v_n) \;=\; f(u_1, \ldots, u_n) \vee f(v_1, \ldots, v_n)$$

Boole-függvény kiértékelésére *kevesebb*, mint $2C(f)$. Ebből pedig következik, hogy a függvény-dekompozíció *nem* mindig jó ötlet.

Legyen $n = m + 2^m$ és vezessük be az $f(i_1, \ldots, i_m, x_0, \ldots, x_{2^m-1}) = f_i(x)$ jelölést, ahol i alatt az $(i_1 \ldots i_m)_2$ számot értjük. Ekkor $(u_1, \ldots, u_n) = (i_1, \ldots, i_m, x_0, \ldots, x_{2^m-1})$, $(v_1, \ldots, v_n) = (j_1, \ldots, j_m, y_0, \ldots, y_{2^m-1})$ és $F(u, v) = f_i(x) \vee f_j(y)$.

a) Bizonyítsuk be, hogy $O(n/\lg n)^2$ költségű lánccal kiértékelhető a következő 2^m+1 függvény:

$$z_l \;=\; x \oplus \big(([l \leq i] \oplus [i \leq j]) \wedge (x \oplus y)\big), \qquad 0 \leq l \leq 2^m,$$

ahol i, j, x, y adott vektorok és valamennyi z_l vektor hossza 2^m.

b) Legyen $g_i(x) = f_i(x) \oplus f_{i-1}(x)$ minden $0 \leq i \leq 2^m$ esetén, ahol $f_{-1}(x) = f_{2^m}(x) = 0$. Becsülje meg a $c_l = g_l(z_l)$ - összesen $2^m + 1$ számú - érték kiszámításának költségét, ha a z_l vektorok adottak minden $0 \leq l \leq 2^m$ esetén.

c) Legyen $c'_l = c_l \wedge ([i \leq j] \equiv [l \leq i])$ és $c''_l = c_l \wedge ([i \leq j] \equiv [j > l])$. Bizonyítsuk be, hogy

$$f_i(x) = c'_0 \oplus c'_1 \oplus \cdots \oplus c'_{2^m}, \qquad f_j(y) = c''_0 \oplus c''_1 \oplus \cdots \oplus c''_{2^m}.$$

d) Mutassa meg, hogy $C(F) \leq 2^n/n + O(2^n(\lg n)/n^2)$. (Ha n eléggé nagy, akkor ez a költség határozottan kisebb mint $2^{n+1}/n$, de léteznek olyan f függvények, amelyekre $C(f) > 2^n/n$.)

e) Írjuk fel az F-hez tartozó láncot, ha $m = 1$ és $f(i, x_0, x_1) = (i \wedge x_0) \vee x_1$.

77. [*35*] (N. P. Redkin, 1970 [43].) Tegyük fel, hogy egy Boole-lánc csak az ÉS, VAGY és NEM műveleteket használja fel, azaz minden lépés $x_i = x_{j(i)} \wedge x_{k(i)}$, $x_i = x_{j(i)} \vee x_{k(i)}$ vagy $x_i = \bar{x}_{j(i)}$. Bizonyítsa be, hogy ha egy ilyen lánc az $f_n(x_1, \ldots, x_n) = x_1 \oplus \cdots \oplus x_n$ „páratlan” vagy az $\bar{f}_n(x_1, \ldots, x_n) = 1 \oplus x_1 \oplus \cdots \oplus x_n$ „páros” függvényt számítja ki $n \geq 2$-re, akkor a lánc hossza legalább $4(n-1)$.

78. [*26*] (W. J. Paul, 1977 [39].) Legyen $f(x_1, \ldots, x_m, y_0, \ldots, y_{2^m-1})$ egy olyan Boole-függvény, ami y_k értékét veszi fel valahányszor $(x_1 \ldots x_m)_2 = k \in S$ teljesül, ahol $S \subseteq \{0, 1, \ldots, 2^m - 1\}$ egy adott halmaz. Az f többi pontban felvett értékéről nem teszünk fel semmit. Mutassuk meg, hogy $C(f) \geq 2|S| - 2$, ha S nem az üres halmaz. (Speciálisan, ha $S = \{0, 1, \ldots, 2^m - 1\}$, a 39. gyakorlatban látott multiplexer lánc aszimptotikusan optimális.)

79. [*32*] (C. P. Schnorr, 1976 [48].) Azt mondjuk, hogy egy u és v változók „társak” egy Boole-láncban, ha pontosan egy egyszerű út létezik közöttük a láncnak megfelelő bináris fában. Két változó csak akkor lehet társ, ha mindegyiküket csak egyszer használtuk fel a láncban, de ez a szükséges feltétel nem elégséges. Például, a 2 és 4 változók barátok az $S_{1,2,3}$-hez tartozó láncban a 7.9. ábrán, de nem társak az S_2-höz tartozó láncban.

a) Bizonyítsuk be, hogy ha egy n változós Boole-láncban nincsenek társak, akkor a lánc költsége legalább $2n - 2$.

b) Bizonyítsuk be, hogy $C(f) = 2n - 3$, ha f az $S_{0,n}(x_1, \ldots, x_n)$ minden-egyenlő függvény.

80. [*M27*] (L. J. Stockmeyer, 1977 [57].) A szimmetrikus függvények egy másik jelölése néha kényelmesebb: Ha $\alpha = a_0 a_1 \ldots a_n$ egy bináris füzér, legyen $S_\alpha(x) = a_{\nu x}$. Például $\langle x_1 x_2 x_3 \rangle = S_{0011}$ és $x_1 \oplus x_2 \oplus x_3 = S_{0101}$ ezzel a jelöléssel. Vegyük észre, hogy $S_\alpha(0, x_2, \ldots, x_n) = S_{\alpha'}(x_2, \ldots, x_n)$ és $S_\alpha(1, x_2, \ldots, x_n) = S_{'\alpha}(x_2, \ldots, x_n)$, ahol α' és $'\alpha$ α-t jelölik az utolsó, illetve az első elemének törlése után. Ugyancsak,

$$S_\alpha\big(f(x_3, \ldots, x_n), \bar{f}(x_3, \ldots, x_n), x_3, \ldots, x_n\big) \;=\; S_{'\alpha'}(x_3, \ldots, x_n)$$

ha f egy tetszőleges Boole-függvény $n - 2$ változóval.

a) Egy paritás függvényre $a_0 \neq a_1 \neq a_2 \neq \cdots \neq a_n$. Tegyük fel, hogy $n \geq 2$. Bizonyítsuk be, hogy ha S_α nem egy paritás függvény és $S_{'\alpha'}$ nem konstans, akkor

$$C(S_\alpha) \;\geq\; \max\big(C(S_{\alpha'})+2,\, C(S_{'\alpha})+2,\, \min\big(C(S_{\alpha'})+3,\, C(S_{'\alpha})+3,\, C(S_{'\alpha'})+5\big)\big).$$

b) Az előző eredményt felhasználva milyen alsó korlát adható $C(S_k)$-ra és $C(S_{\geq k})$-ra, ha $0 \leq k \leq n$?

81. [*23*] (M. Snir, 1986 [53].) Mutassuk meg, hogy minden c költségű és d mélységű láncra, ami a 36. gyakorlatban látott prefix problémát megoldja, $c + d \geq 2n - 2$ teljesül.

82. [*M23*] Magyarázzuk meg a (62)–(70) logikai állításokat. Melyek igazak közölük?

83. [*21*] Mutassuk meg, hogy ha egy $f(x_1, \ldots, x_n)$-et kiszámító Boole-lánc p kényszerítő műveletet tartalmaz, akkor $C(f) < (p+1)(n + p/2)$.

84. [*M20*] Egy *monoton Boole-lánc* egy olyan Boole-lánc, amiben minden $\circ_i$ művelet monoton. f legrövidebb monoton láncának hosszát $C^+(f)$-fel jelöljük. Mutassuk meg, hogy ha létezik $f(x_1, \ldots, x_n)$-re olyan monoton Boole-lánc, ami p $\wedge$ és q $\vee$ műveletet tartalmaz, akkor $C^+(f) < \min\big((p+1)(n + p/2), (q+1)(n + q/2)\big)$.

85. [*M28*] Jelölje M_n az n-változós monoton függvények halmazát. Ha L M_n-beli függvények egy családja, legyen

$$x \sqcup y = \bigwedge\{z \in L \mid z \supseteq x \vee y\} \qquad \text{és} \qquad x \sqcap y = \bigvee\{z \in L \mid z \subseteq x \wedge y\}.$$

Az L családot ekkor „legitimnek” nevezzük, ha tartalmazza a 0 és az 1 konstans függvényeket, az x_j projekciókat $1 \leq j \leq n$-re, továbbá ha $x \sqcup y \in L$, $x \sqcap y \in L$ teljesül minden $x, y \in L$ esetén.

a) Ha $n = 3$, akkor $M_3 = \{\texttt{00}, \texttt{01}, \texttt{03}, \texttt{05}, \texttt{11}, \texttt{07}, \texttt{13}, \texttt{15}, \texttt{0f}, \texttt{33}, \texttt{55}, \texttt{17}, \texttt{1f}, \texttt{37}, \texttt{57}, \texttt{3f}, \texttt{5f}, \texttt{77}, \texttt{7f}, \texttt{ff}\}$, ahol minden függvényt a hexadecimális igazságtáblája reprezentál. Összesen 2^{15}-féleképpen lehet az L függvénycsaládot megválasztani úgy, hogy $\{\texttt{00}, \texttt{0f}, \texttt{33}, \texttt{55}, \texttt{ff}\} \subseteq L \subseteq M_3$ teljesüljön. Hány lesz ezek közül legitim?

b) Ha A egy részhalmaza az $\{1, \ldots, n\}$ halmaznak, akkor legyen $\lceil A \rceil = \bigvee_{a \in A} x_a$, továbbá legyen $\lceil \infty \rceil = 1$. Tegyük fel, hogy $\mathcal{A}$ egy olyan családja $\{1, \ldots, n\}$ részhalmazainak, hogy tartalmazza az összes legfeljebb egy elemű részhalmazt és zárt a metszetre, azaz $A \cap B \in \mathcal{A}$ minden $A \in \mathcal{A}$, $B \in \mathcal{A}$ esetén. Bizonyítsuk be, hogy az $L = \{\lceil A \rceil \mid A \in \mathcal{A} \cup \{\infty\}\}$ család legitim.

c) Legyen $(x_{n+1}, \ldots, x_{n+r})$ egy monoton Boole-lánc 1. Legyen $(\hat{x}_{n+1}, \ldots, \hat{x}_{n+r})$ az a Boole-lánc, amit úgy kapunk, hogy az előző láncban minden $\wedge$-t $\sqcap$ műveleti jellel és minden $\vee$-ot $\sqcup$ művelettel cserélünk fel, ahol most L egy rögzített legitim család. Bizonyítsuk be, hogy ha $n+1 \le l \le n+r$, akkor

$$\hat{x}_l \ \subseteq \ x_l \vee \bigvee_{i=n+1}^{l} \{\hat{x}_i \oplus (\hat{x}_{j(i)} \vee \hat{x}_{k(i)}) \mid \circ_i = \vee\};$$

$$x_l \ \subseteq \ \hat{x}_l \vee \bigvee_{i=n+1}^{l} \{\hat{x}_i \oplus (\hat{x}_{j(i)} \wedge \hat{x}_{k(i)}) \mid \circ_i = \wedge\}.$$

86. [*F37*] Egy G gráf az $\{1, \ldots, n\}$ csúcsok felett megadható $N = \binom{n}{2}$ számú x_{uv}, $1 \le u < v \le n$ Boole-változó segítségével, ahol $x_{uv} = [u — v\ G\text{-ben van}]$. Legyen f az $f(x) = [G$ tartalmaz háromszöget$]$ függvény. Például, ha $n = 4$, akkor $f(x_{12}, x_{13}, x_{14}, x_{23}, x_{24}, x_{34}) = (x_{12} \wedge x_{13} \wedge x_{23}) \vee (x_{12} \wedge x_{14} \wedge x_{24}) \vee (x_{13} \wedge x_{14} \wedge x_{34}) \vee (x_{23} \wedge x_{24} \wedge x_{34})$. Ebben a gyakorlatban az a célunk, hogy bebizonyítsuk, a $C^+(f)$ monoton bonyolultság éppen $\Omega(n/\log n)_3$.

a) Ha egy G gráfban $u_j — v_j$ $1 \le j \le r$ esetén, akkor nevezzük az $S = \{\{u_1, v_1\}, \ldots, \{u_r, v_r\}\}$ halmazt egy r-*családnak*, és legyen $\Delta(S) = \bigcup_{1 \le i < j \le r} (\{u_i, v_i\} \cap \{u_j, v_j\})$ a páronként vett metszetek elemeinek halmaza. Egy G gráfot nevezzünk r-*zártnak*, ha $u — v$ mindig teljesül $\Delta(S) \subseteq \{u, v\}$ esetén bármely S r-családra. A gráfot *erősen* r-zártnak nevezzük, ha a fentieken túl még az is igaz, hogy $|\Delta(S)| \ge 2$ minden S r-családra. Bizonyítsa be, hogy egy erősen r-zárt gráf erősen $(r+1)$-zárt is.

b) Bizonyítsuk be, hogy a $K_{m,n}$ teljes páros gráf erősen r-zárt, ha $r > \max(m, n)$.

c) Bizonyítsuk be, hogy egy erősen r-zárt gráfnak legfeljebb $(r-1)^2$ éle lehet.

d) Legyen L a következő függvények családja: $\{1\} \cup \{\lceil G \rceil \mid G$ egy erősen r-zárt gráf az $\{1, \ldots, n\}$ csúcsok felett$\}$. (Lásd még 85(b) gyakorlat; G-t élek halmazának tekintjük. Például, ha az élek $1 — 3$, $1 — 4$, $2 — 3$ és $2 — 4$, akkor $\lceil G \rceil = x_{13} \vee x_{14} \vee x_{23} \vee x_{24}$.) Legitim lesz-e L?

e) Legyen $x_{N+1}, \ldots, x_{N+p+q} = f$ egy monoton Boole-lánc p $\wedge$-lépéssel és q $\vee$-lépéssel, és tekintsük a a (d) rész L családja szerint módosított $\hat{x}_{N+1}, \ldots, \hat{x}_{N+p+q} = \hat{f}$ láncot. Ha $\hat{f} \ne 1$, mutassa meg, hogy $2(r-1)^3 p + (r-1)^2 (n-2) \ge \binom{n}{3}$. *Útmutatás:* Használjuk a 85(c) gyakorlatban szereplő második formulát.

f) Továbbá, ha $\hat{f} = 1$, akkor $r^2 q \ge 2^{r-1}$.

g) Következésképpen $p = \Omega(n/\log n)_3$. *Útmutatás:* Legyen $r \approx 6 \lg n$ és alkalmazzuk a 84. gyakorlatot.

87. [*M20*] Mutassuk meg, hogy ha nemmonoton műveletek is megengedettek, akkor a 86. gyakorlatban szereplő háromszög-függvény költsége $C(f) = O(n^{\lg 7} (\lg n)^2) = O(n^{2.81})$. *Útmutatás:* Egy gráf pontosan akkor tartalmaz háromszöget, ha a gráf szomszédsági mátrixának a kockája nemnulla átlójú.

88. [*40*] Egy *medián-lánc* olyan, mint egy Boole-lánc, csak a bináris műveletek helyett háromelem-mediánja lépésekből áll, azaz $x_i = \langle x_{j(i)} x_{k(i)} x_{l(i)} \rangle$, ha $n+1 \le i \le n+r$.

Vizsgáljuk meg a medián láncok optimális hosszának, mélységének és költségének kérdéseit a 7 változós önduális monoton Boole-függvények esetében. Mi a legrövidebb lánc az $\langle x_1x_2x_3x_4x_5x_6x_7\rangle$ függvényre?

A 7.1.2. pont gyakorlatainak megoldása

1. $((x_1 \vee x_4) \wedge x_2) \equiv (x_1 \vee x_3)$.

2. (a) $(w \oplus (x \wedge y)) \oplus ((x \oplus y) \wedge z)$; (b) $(w \wedge (x \vee y)) \wedge ((x \wedge y) \vee z)$.

3. [18] Tekintsünk egy $k \times n$-es mátrixot, melynek sorait azon x vektorok alkotják, melyekre $f(x) = 1$. A változók permutálásával és/vagy negálásával feltételezhetjük, hogy az első sor az $1 \ldots 1$ és az oszlopok rendezve vannak. Tegyük fel, hogy l különböző oszlopunk van. Ekkor $f = g \wedge h$, melyben g az összes $(x_{j-1} \equiv x_j)$ kifejezés konjunkciója, ahol $1 < j \leq n$, és a $j-1$. oszlop megegyezik a j. oszloppal, míg h jelöli a k darab l hosszúságú minterm diszjunkcióját, ahol az azonos oszlopok közül mindig csak egyet használunk. Például, ha $n = 8$ és f a következő $k = 3$ pontban igaz: 11111111, 00001111, 00110111, akkor $l = 4$, míg $f(x)$ megegyezik $(x_1 \equiv x_2) \wedge (x_3 \equiv x_4) \wedge (x_6 \equiv x_7) \wedge (x_7 \equiv x_8) \wedge ((x_1 \wedge x_3 \wedge x_5 \wedge x_6) \vee (\bar{x}_1 \wedge \bar{x}_3 \wedge x_5 \wedge x_6) \vee (\bar{x}_1 \wedge x_3 \wedge \bar{x}_5 \wedge x_6))$ formulával. Az így kapott formula hossza $2n + (k-2)\,l - 1$, ahol $l \leq 2^{k-1}$.

Figyeljük meg, hogy ha k nagy, akkor rövidebb formulát kapunk, ha $f(x)$-et $f_1(x) \vee \ldots \vee f_r(x)$ alakban írjuk fel, ahol minden f_j legfeljebb $\lceil k/r \rceil$ 1-est tartalmaz. Ekkor

$$L(f) \leq \min_{r \geq 1} \left(r - 1 + \left(2n + \lceil k/r - 2 \rceil \, 2^{\lceil k/r - 1 \rceil} \right) r \right).$$

4. Az első egyenlőtlenség közvetlen következménye annak, hogy egy d mélységű bináris fa legfeljebb $1 + 2 + \ldots + 2^{d-1} = 2^d - 1$ belső csúcsot tartalmazhat.

Az utalás rendben van, hiszen létezik legkisebb részfa, melynek mérete $\geq \lfloor r/3 \rfloor$. Ennek s mérete legfeljebb $1 + 2\,(\lfloor r/3 \rfloor - 1)$. Ez alapján $f = (g?\ f_1 :\ f_0)$, ahol g egy s méretű részformula, míg f_0 és f_1 olyan $r - s - 1$ méretű formulák, melyeket f-ből a g helyére rendre 0, illetve 1 helyettesítésével kapunk.

Legyen $d(r) = \max\{D(f) \mid L(f) = r\}$. Mivel a mux függvény mélysége 2, és $\max(s, r-s-1) < \lceil \frac{2r}{3} \rceil$, ezért $r \geq 3$ esetén $d(r) \leq 2 + d\left(\lceil \frac{2r}{3} \rceil - 1\right)$, melyből az eredmény r szerinti indukcióval következik [54].

5. Legyen $g_0 = 0$, $g_1 = x_1$, és $g_j = x_j \wedge (x_{j-1} \vee g_{j-2})$, ahol $j \geq 2$. Ekkor $F_n = g_n \vee g_{n-1}$, melynek költsége $2n - 2$, mélysége pedig n. [Ezek a g_j függvények fontos szerepet játszanak a bináris összeadásnál is. A 42., illetve 44. gyakorlatoknál bemutatjuk, hogyan lehet őket $O(\lg n)$ mélységű függvényekkel meghatározni.]

6. Igaz: vizsgálja meg az $y = 0$ valamint az $y = 1$ esetet.

7. $\hat{x}_5 = x_1 \vee x_4$, $\hat{x}_6 = x_2 \wedge \hat{x}_5$, $\hat{x}_7 = x_1 \vee x_3$, $\hat{x}_8 = \hat{x}_6 \oplus \hat{x}_7$. (Az eredeti lánc a (6)-os „random”függvényt számítja; lásd az 1. gyakorlatot. Az új lánc ennek a függvénynek a normalizáltját, nevezetesen a negáltját számítja.)

8. Az x_k igazságtáblájában 2^{n-k} darab 0-t, valamint 2^{n-k} darab 1-est tartalmazó blokkok váltakoznak, amint az a (7)-ben is megfigyelhető. így ha megszorozzuk $2^{2^{n-k}} + 1$-gyel, az eredményül kapott $x_k + \left(x_k \ll 2^{n-k}\right)$ csupa 1-esből fog állni.

9. Az L6-os lépésben, amikor $L(f) = \infty$, tároljuk el g-t és h-t egy f-hez rendelt rekordban. Ekkor f-hez egy rekurzív eljárássegítségével, a megfelelő g és h felhasználásával előállítható egy minimum hosszúságú formula.

10. Az L3-as lépésben írjunk $k = r - 1 - j$ helyére $k = r - 1$-et, valamint írjuk át mindenhol az L-et D-re.

11. Az egyetlen probléma az, hogy az U3 lépésben *csökkenteni* kell j-t. Ekkor $j = 0$ esetén $\phi(g) \,\&\, \phi(h) \neq 0$, így az összes $r-1$ költségű esetet megtaláljuk, még mielőtt nekifognánk az $r-1$. listának.

U1. Kezdeti beállítás. Legyen $U(0) \leftarrow \phi(0) \leftarrow 0$ míg $U(f) \leftarrow \infty$ minden $1 \le f < 2^{2^n-1}$ esetén. Ekkor legyen $U(x_k) \leftarrow \phi(x_k) \leftarrow 0$ és fűzzük x_k-t a 0. listához, ahogy azt az L1 lépésben tettük. Ugyanakkor legyen $U(x_j \circ x_k) \leftarrow 1$, és rendeljük $\phi(x_j \circ x_k)$-hoz a neki megfelelő 1 súlyú bit vektort, és fűzzük $x_j \circ x_k$-t az 1. listához, minden $1 \le j < k \le n$ és mind az öt $\circ$ normál operátor esetén. Végül legyen $c \leftarrow 2^{2^n-1} - 5\binom{n}{2} - n - 1$.

U2. Ciklus r-re. Ismételjük az U3 lépést $r = 2,\ 3,\ \ldots$ esetén, amíg $c > 0$.

U3. Ciklus j-re és k-ra. Ismételjük az U4 lépést $j = \lfloor (r-1)/2 \rfloor,\ \lfloor (r-1)/2 \rfloor - 1,\ \ldots$, valamint $k = r - 1 - j$ esetén, amíg $j \ge$.

U4. Ciklus g-re és h-ra. Ismételjük az U5 lépést a j. lista minden g és a k. lista minden h elemére. Amennyiben $j = k$, csak azon h elemeket tekintsük, melyek a g *után következnek* a k. listán.

U5. Ciklus f-re.

Ha $\phi(g) \,\&\, \phi(h) \neq 0$, akkor legyen $u \leftarrow r - 1$ és $v \leftarrow \phi(g) \,\&\, \phi(h)$, különben pedig legyen $u \leftarrow r$ és $v \leftarrow \phi(g) \,@|@\, \phi(h)$. Ekkor végezzük el az U6 lépést $f = g \,\&\, h$, $f = \bar{g} \,\&\, h$, $f = g \,\&\, \bar{h}$, $f = g \,@|@\, h$, illetve $f = g \oplus h$ esetén.

U6. $U(f)$ és $\phi(f)$ aktualizálása. Ha $U(f) = \infty$, akkor legyen $c \leftarrow c - 1$, $\phi(f) \leftarrow v$, és fűzzük f-et az u. listához. Különben, ha $U(f) > u$, akkor legyen $\phi(f) \leftarrow v$, és helyezzük át f-et az $U(f)$. listáról az u. listára. Ha pedig $U(f) = u$, akkor legyen $\phi(f) \leftarrow \phi(f) \,@|@\, v$. ■

12. $x_4 = x_1 \oplus x_2$, $x_5 = x_3 \wedge x_2$, $x_6 = x_2 \wedge \bar{x}_4$, $x_7 = x_5 \vee x_6$.

13. $f_5 = 01010101$ (x_3); $f_4 = 01110111$ $(x_2 \vee x_3)$; $f_3 = 01110101$ $((\bar{x}_1 \wedge x_2) \vee x_3)$; $f_2 = 00110101$ $(x_1?\ x_3\colon x_2)$; $f_1 = 00010111$ $(\langle x_1 x_2 x_3 \rangle)$.

14. Tetszőleges $1 \le j \le n$ esetén számítsuk ki $t \leftarrow (g \oplus (g \gg 2^{n-j})) \,\&\, x_j$, $t \leftarrow t \oplus (t \ll 2^{n-j})$, ahol x_j a (11)-es igazságtábla. Ekkor $1 \le k \le n$ és $k \neq j$ esetén az $x_j \leftarrow x_j \circ x_k$ műveletnek megfelelő igazságtábla a $g \oplus (t \,\&\, ((x_j \circ x_k) \oplus x_j))$ képlettel számítható. (Mivel az $(x_j \circ x_k) \oplus x_j$ által leírt $5n(n-1)$ különböző művelet g-től független, azok előre kiszámíthatóak. Ugyanez az ötlet működik az ennél általánosabb $x_{j(i)} \leftarrow x_{k(i)} \circ_i x_{l(i)}$ alakú számításokra, és a neki megfelelő $(x_k \circ x_l) \oplus x_j$ által leírt $5n^2(n-1)$ féle műveletre is.)

15. Valódi aszimmetrikus módszerek szimmetrikus függvények kiszámítására:

(a)	(b)	(c)
$x_1 \leftarrow x_1 \oplus x_2$,	$x_1 \leftarrow x_1 \oplus x_2$,	$x_1 \leftarrow x_1 \oplus x_2$,
$x_3 \leftarrow x_3 \oplus x_4$,	$x_2 \leftarrow x_2 \wedge \bar{x}_1$,	$x_2 \leftarrow x_2 \oplus x_3$,
$x_1 \leftarrow x_1 \oplus x_3$,	$x_3 \leftarrow x_3 \oplus x_4$,	$x_2 \leftarrow x_2 \vee x_1$,
$x_2 \leftarrow x_2 \oplus x_4$,	$x_4 \leftarrow x_4 \wedge x_1$,	$x_1 \leftarrow x_1 \oplus x_4$,
$x_3 \leftarrow x_3 \vee x_2$,	$x_2 \leftarrow \bar{x}_2 \wedge x_3$,	$x_1 \leftarrow x_1 \wedge x_3$,
$x_3 \leftarrow x_3 \wedge \bar{x}_1$.	$x_2 \leftarrow x_2 \oplus x_1$,	$x_2 \leftarrow x_2 \wedge \bar{x}_1$,
	$x_2 \leftarrow x_2 \wedge \bar{x}_4$.	$x_2 \leftarrow x_2 \oplus x_4$.

16. Az a számítás, amely csak $\oplus$-t és komplementálást (negációt) tartalmaz nem vezet sehova kivéve az affin függvényeket (lásd a **??**. gyakorlatot.) Tegyük fel, hogy $f(x) = f(x_1, \ldots, x_n)$ egy minimális memóriával kiszámítható nem affin függvény. Ekkor $f(x)$ alakja $g(Ax + c)$, ahol $g(y_1, y_2, \ldots, y_n) = g(y_1 \wedge y_2, y_2, \ldots, y_n)$, bizonyos nem szinguláris $n \times n$-es 0-ákból és 1-esekből álló A mátrixra, ahol x és c oszlopvektorok, és a vektorműveleteket modulo 2 végezzük el. Ebben a formulában az A mátrix és a c vektor biztosítják az összes $x_i \leftarrow x_i \oplus x_j$ műveletet és/vagy a permutációkat és azon koordináták komplementálását, amelyek a legutolsó végrehajtott nem-affin művelet után jelennek meg. Ki fogjuk használni azt a tényt, hogy $g(0,0,y_3,\ldots,y_n) = g(1,0,y_3,\ldots,y_n)$.

Legyen α és β az A első két sora, valamint legyen a és b a c első két eleme. Ha $Ax + c \equiv y$ (modulo 2), akkor $y_1 = y_2 = 0$ akkor és csak akkor, ha $\alpha{\cdot}x \equiv a$ és $\beta{\cdot}x \equiv b$. Ezt a feltételt pontosan 2^{n-2} x vektor teljesíti, és minden ilyen vektorra $f(x) = f(x \oplus w)$, ahol $Aw \equiv (1,0,\ldots,0)^T$.

Adott α, β, a, b, és w-re, $\alpha \neq (0,\ldots,0)$, $\beta \neq (0,\ldots,0)$, $\alpha \neq \beta$, és $\alpha \cdot w \equiv 1$ (modulo 2) esetén, létezik $2_{2_n-2_{n-2}}$ darab olyan f függvény amely az $f(x) = f(x \oplus w)$ amikor $\alpha \cdot x \mod 2 = a$ és $\beta \cdot x \mod 2 = b$ tulajdonsággal rendelkezik. Ennélfogva a minimális memóriában kiszámítható függvények száma legfeljebb 2^{n+1} (affin függvényekre) plusz

$$(2^n-1)(2^n-2)2^2(2^{n-1})(2^{2^n-2^{n-2}}) < 2^{2^n-2^{n-2}+3n+1}.$$

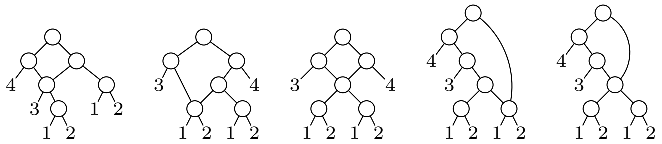
17. Legyen $f(x_1,\ldots,x_n) = g(x_1,\ldots,x_{n-1}) \oplus (h(x_1,\ldots,x_{n-1}) \wedge x_n)$ mint a (**??**)-ban. Reprezentáljuk h-t KNF-ben és $h \wedge x_n$ eléréséhez tegyük a klózokat egyenként x_0-ba és x_n-el vett AND-jüket tegyük x_n-be. Állítsuk elő g-t mint konjunkciók (mod 2) vett összegét, egymásután berakva a konjunkciókat x_0-ba és XOR-jukat pedig x_n-be.

(Nem lehet minden függvényt $n+1$ regiszterben kiértékelni, ha nem engedjük meg a $\oplus$ és a $\equiv$ - nem irányító - műveletek használatát. Azonban világos, hogy $n+2$ regiszter már elegendő, még akkor is, ha egyedül a $\overline{\wedge}$ műveletet használjuk.)

18. Ahogy a 14-es megoldásban említettük a minimális memóriával való kiszámítás szöveges definícióját ki kell hogy terjesszük, megengedve az olyan lépéseket, mint $x_{j(i)} \leftarrow x_{k(i)} \circ_i x_{@l(i)}$, ahol $k(i) \neq j(i)$ és $l(i) \neq j(i)$. Ugyanis ezek jobb eredményeket adnak bizonyos olyan függvényekre, amelyek az öt változójuk közül csak négytől függnek. Ezután azt találtuk, hogy $C_m(f) = (0,1,\ldots,13,14)$ rendre (2, 2, 5, 20, 93, 389, 1960, 10459, 47604, 135990, 198092, 123590, 21540, 472, 0) függvényosztályra ... amiben nincs benne 75,908 osztály (és 575,963,136 függvény) amelyekre $C_m(f) = \infty$, miután egyáltalán nem értékelhetők ki minimális memóriában. A legérdekesebb ilyen jellegű függvény talán $(x_1 \wedge x_2) \vee (x_2 \wedge x_3) \vee (x_3 \wedge x_4) \vee (x_4 \wedge x_5) \vee (x_5 \wedge x_1)$, amelyre $C(f) = 7$ de $C_m(f) = \infty$. Egy másik érdekes eset $\big(((x_1 \vee x_2) \oplus x_3) \vee ((x_2 \vee \bar{x}_4) \wedge x_5)\big) \wedge \big((x_1 \equiv x_2) \vee x_3 \vee x_4\big)$, amelyre $C(f) = 8$ és $C_m(f) = 13$. Egy lehetőség e függvény 8-lépéses kiértékelésére $x_6 = x_1 \vee x_2$, $x_7 = x_1 \vee x_4$, $x_8 = x_2 \oplus x_7$, $x_9 = x_3 \oplus x_6$, $x_{10} = x_4 \oplus x_9$, $x_{11} = x_5 \vee x_9$, $x_{12} = x_8 \wedge x_{10}$, $x_{13} = x_{11} \wedge \bar{x}_{12}$.

19. Ha nincs, akkor a gyökér bal- és jobb oldali részfáiban kell hogy fedés legyen, mivel az (i) eset nem teljesül. A hipotézis szerint minden változónak legalább egyszer kell levélben szerepelni. Mivel az (ii) eset nem teljesül legalább két változónak legalább kétszer levélben szerepelnie. Azonban a belső csúcsok száma $r \leq n+1$ és így a levelek száma nem lehet $n+2$, hacsak a részfák között nincs átfedés.

20. Az L algoritmus mutatja (ha elhagyjuk „$f = g \oplus h$”-t az L5 lépésben), hogy bizonyos formulák hossza kell hogy 15 legyen, és sajnos a 11 gyakorlatbeli lábnyom módszer sem ad 14-nél jobb eredményt. Valóban minimális lánc elérése érdekében a szövegben bemutatott 25 darab $r = 6$-ra vonatkozó speciális láncot ki kell egészíteni öt olyan másikkal, amelyek nem hagyhatók ki, nevezetesen



és amikor $r = (7,8,9)$ rendre meg kell vizsgálni $(653, 12387, 225660)$ pótlólagos potenciális láncokat, amelyek nem speciális esetei a fentről-lefele és a lentről-felfele konstrukcióknak. Az 7.6.

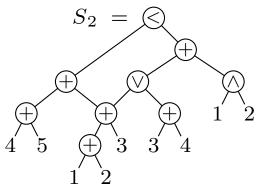
táblázattal való összehasonlítás eredményét mutatja a következő táblázat.

$C_c(f)$	Osztá-lyok	Függ-vények	$U_c(f)$	Osztá-lyok	Függ-vények	$L_c(f)$	Osztá-lyok	Függ-vények	$D_c(f)$	Osztá-lyok	Függ-vények
0	2	10	0	2	10	0	2	10	0	2	10
1	1	48	1	1	48	1	1	48	1	1	48
2	2	256	2	2	256	2	2	256	2	7	684
3	7	940	3	7	940	3	7	940	3	59	17064
4	9	2336	4	9	2336	4	7	2048	4	151	47634
5	24	6464	5	21	6112	5	20	5248	5	2	96
6	30	10616	6	28	9664	6	23	8672	6	0	0
7	61	18984	7	45	15128	7	37	11768	7	0	0
8	45	17680	8	40	14296	8	27	10592	8	0	0
9	37	7882	9	23	8568	9	33	11536	9	0	0
10	4	320	10	28	5920	10	16	5472	10	0	0
11	0	0	11	6	1504	11	30	6304	11	0	0
12	0	0	12	5	576	12	3	960	12	0	0
13	0	0	13	3	144	13	8	1472	13	0	0
14	0	0	14	2	34	14	2	96	14	0	0
15	0	0	15	0	0	15	4	114	15	0	0

A két 5 mélységű függvényosztályt az $S_{2,4}(x_1,x_2,x_3,x_4)$ és az $x_1 \oplus S_2(x_2,x_3,x_4)$ szimmetrikus függvények reprezentálják, és ennek a két függvénynek az $S_2(x_1,x_2,x_3,x_4)$ és az $S_{1,3}(x_1,x_2,x_3,x_4) = x_1 \oplus x_2 \oplus x_3 \oplus x_4$ paritás függvénnyel együtt a hossza 15. Úgyszintén $U_c(S_{2,4}) = U_c(S_{1,3}) = 14$. Hasonlóan reprezentálják a négy 10 költségű osztályt az $S_{1,4}(x_1,x_2,x_3,x_4)$, $S_{2,4}(x_1,x_2,x_3,x_4)$, $(x_4?\ x_1 \oplus x_2 \oplus x_3\colon \langle x_1x_2x_3\rangle)$, és $[(x_1x_2x_3x_4)_2 \in \{0,1,4,7,10,13\}]$. (Ezek közül a harmadik, mellékesen komplemense a (20)-nak, a legnehezebb eset a „Harvard"-on.)

21. (A szerzők állítása szerint az ő táblabemeneteik „az akkori írók által ismert leggazdaságosabb műveleteket kellett hogy tekintsék".)

22. $\nu(x_1x_2x_3x_4x_5) = 3$ akkor és csak akkor, ha $\nu(x_1x_2x_3x_4) \in \{2,3\}$ és $\nu(x_1x_2x_3x_4x_5)$ páratlan. Hasonlóan, $S_2(x_1,x_2,x_3,x_4,x_5) = S_3(\bar{x}_1,\bar{x}_2,\bar{x}_3,\bar{x}_4,\bar{x}_5)$ magában foglalja $S_{1,2}(x_1,x_2,x_3,x_4)$-et:



23. Csak 32 normalizált esetet kell tekinteni mint a 9. ábrán, mivel egy szimmetrikus függvény komplementere is szimmetrikus. Ezután használhatjuk az $S_{1,2}(x) = S_{3,4}(\bar{x})$-hez hasonló tükrözést, lehetőség szerint az $S_{2,3,4,5}(x) = \bar{S}_{0,1}(x) = \bar{S}_{4,5}(\bar{x})$-hez hasonló komplementálással együtt, a megmaradó esetek levezetésére. Természetesen S_1, $S_{1,3,5}$, és $S_{1,2,3,4,5}$ költsége 4. Ami ezután marad $S_{1,2,3,4}(x_1,x_2,x_3,x_4,x_5) = (x_1 \oplus x_2) \vee (x_2 \oplus x_3) \vee (x_3 \oplus x_4) \vee (x_4 \oplus x_5)$, amit a 79. gyakorlatban tárgyaltunk általában n-re.

24. Ahogy a szövegben szerepel, ez a sejtés $n \le 5$-re teljesül.

25. A nem triviális normalizált függvények száma $2^{2^n-1} - n - 1$. (Egy tetszőleges r hosszúságú normalizált láncban, ami nem tartalmazza az összes ilyen függvényt, az $x_j \circ x_k$ egy új függvény lesz j-re és k-ra, ahol $1 \le j,k \le n+r$ és $\circ$ valamilyen normalizált művelet; így minden új lépésben egy új függvényt számíthatunk ki, amíg mindet sorra nem vettük.)

26. Hamis. Például, ha $g = S_{1,3}(x_1, x_2, x_3)$ és $h = S_{2,3}(x_1, x_2, x_3)$, akkor $C(gh) = 5$ a költsége egy teljes összeadónak. De $f = S_{2,3}(x_0, x_1, x_2, x_3)$ költsége 6 a 7.9. ábra szerint.

27. Igen: Az '$x_2 \leftarrow x_2 \oplus x_1$, $x_1 \leftarrow x_1 \oplus x_3$, $x_1 \leftarrow x_1 \wedge \bar{x}_2$, $x_1 \leftarrow x_1 \oplus x_3$, $x_2 \leftarrow x_2 \oplus x_3$' műveletek (x_1, x_2, x_3)-at (z_1, z_0, x_3)-ba képezik le.

28. Legyen $v' = v'' = v \oplus (x \oplus y)$, $u' = ((v \oplus y) \overline{\subset} (x \oplus y)) \oplus u$, $u'' = ((v \oplus y) \vee (x \oplus y)) \oplus u$. Ekkor legyen $u_0 = 0$, $v_0 = x_1$, $u_j = ((v_{j-1} \oplus x_{2j+1}) \vee (x_{2j} \oplus x_{2j+1})) \oplus u_{j-1}$, ha j páratlan, és $u_j = ((v_{j-1} \oplus x_{2j+1}) \overline{\subset} (x_{2j} \oplus x_{2j+1})) \oplus u_{j-1}$, ha j páros, továbbá $v_j = v_{j-1} \oplus (x_{2j} \oplus x_{2j+1})$. Így $(u_j v_j)_2 = (x_1 + \cdots + x_{2j+1}) \mod 4$ minden $1 \le j \le \lfloor n/2 \rfloor$-re. Legyen $x_{n+1} = 0$, ha n páros. Ekkor az $[(x_1 + \cdots + x_n) \mod 4 = 0] = \bar{u}_{\lfloor n/2 \rfloor} \wedge \bar{v}_{\lfloor n/2 \rfloor}$ függvény $\lfloor 5n/2 \rfloor - 2$ lépésben kiszámítható.

Ez a konstrukció L. J. Stockmeyernek köszönhető, aki azt is bebizonyította, hogy ez közel optimális. Valójában a 80. gyakorlat eredménye és a 7.9. és 7.10. ábrák együtt azt mutatják, hogy ez a megoldás legalább egy lépéssel hosszabb a legjobb láncnál minden $n \ge 5$ esetén.

Egyébként az analóg $u''' = ((v \oplus y) \wedge (x \oplus y)) \oplus u$ formula $(u'''v')_2 = ((uv)_2 + x - y) \mod 4$-et eredményezi. Az egyszerűbbnek tűnő $((uv)_2 + x + y) \mod 4$ függvény költsége azonban 6, és nem 5.

29. Hogy felső korlátot kapjunk, tegyük fel, hogy minden teljes és fél összeadó 3-mal növeli meg a mélységet. Ha a_{jd} bit súlya 2^j és mélysége $3d$, akkor legfeljebb $\lceil a_{jd}/3 \rceil$ rákövetkező bit súlya lehet 2^j vagy 2^{j+1} $3(d+1)$ mélység mellett. Ekkor indukcióval adódik, hogy $a_{jd} \le \binom{d}{j} 3^{-d} n + 4$. Így $a_{jd} \le 5$, ha $d \ge \lg_{3/2} n$, és a teljes mélység legfeljebb $3 \log_{3/2} n + 3$. (Érdekes módon a tényleges mélység éppen 100-nak adódik, ha $n = 10^7$.)

30. Ezúttal is jelölje νn a bitek tömörítő összegét n bináris reprezentációjában. Ekkor $s(n) = 5n - 2\nu n - 3\lfloor \lg n \rfloor - 3$.

31. A tömörítő összegek $s(n) < 5n$ lépésben való kiszámítása után, $(z_{\lfloor \lg n \rfloor}, \ldots, z_0)$ egy tetszőleges függvénye kiértékelhető legfeljebb $\sim 2n/\lg n$ lépésben az L tétel szerint. Lásd még [33].

32. Bootstrap: Először bizonyítsuk be n szerinti indukcióval, hogy $t(n) \le 2^{n+1}$.

33. Hamis, egy technikai sajátosság miatt: Ha mondjuk $N = \sqrt{n}$, akkor legalább n lépésre van szükség. A helyes aszimptotikus $N + O(\sqrt{N}) + O(n)$ formulát viszont be lehet bizonyítani. Először is vegyük észre, hogy a fejezet módszerével $N + O(\sqrt{N})$ adódik, ha $N \ge 2^{n-1}$. Különben pedig, ha $\lfloor \lg N \rfloor = n - k - 1$, akkor $O(n)$ művelet segítségével képezhetjük az $\bar{x}_1 \wedge \cdots \wedge \bar{x}_k$ érték és a többi $x_{k+1}, \ldots, x_n$ változó konjunkcióját, majd csökkenthetjük n értékét k-val.

(Ennek az egyik következménye, hogy az $\{S_1, S_2, \ldots, S_n\}$ szimmetrikus függvényeket $s(n) + n + O(\sqrt{n}) = 6n + O(\sqrt{n})$ költséggel és $O(\lg n)$ mélységgel kiszámíthatjuk.)

34. Egy *kiterjesztett* prioritás kódolónak legyen $n+1 = 2^m$ bemenete - $x_0 x_1 \ldots x_n$ - és $m+1$ kimenete - $y_0 y_1 \ldots y_m$ -, ahol $y_0 = x_0 \vee x_1 \vee \cdots \vee x_n$. Ha Q'_m és Q''_m kiterjesztett kódolók $x'_0 \ldots x'_n$, illetve $x''_0 \ldots x''_n$ bemenettel, akkor Q_{m+1} $x'_0 \ldots x'_n x''_0 \ldots x''_n$ bemenettel a következőféleképpen konstruálható meg a segítségükkel: $y_0 = y'_0 \vee y''_0$, $y_1 = y''_0$, $y_2 = y_1?\ y''_1\colon y'_1$, $\ldots$, $y_{m+1} = y_1?\ y''_m\colon y'_m$. Ha P'_m egy közönséges prioritás kódoló $x'_1 \ldots x'_n$ bemenettel, akkor P_{m+1} $x'_1 \ldots x'_n x''_0 \ldots x''_n$ bemenettel hasonlóféleképpen adható meg.

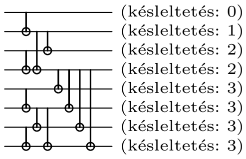
Kezdve $m = 2$-vel $y_2 = x_3 \vee (x_1 \wedge \bar{x}_2)$, $y_1 = x_2 \vee x_3$ és $y_0 = x_0 \vee x_1 \vee y_1$ adódik. A konstrukció egy p_m költségű P_m-et és egy q_m költségű Q_m-et eredményez, ahol $p_2 = 3$, $q_2 = 5$, továbbá $p_{m+1} = 3m + p_m + q_m$ és $q_{m+1} = 3m + 1 + 2q_m$, ha $m \ge 2$. Így tehát $p_m = q_m - m$ és $q_m = 15 \cdot 2^{m-2} - 3m - 4 \approx 3.75n$.

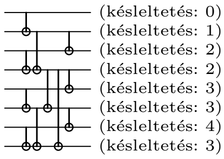
35. Ha $n = 2m$, akkor számítsuk ki $x_1 \wedge x_2$, $\ldots$, $x_{n-1} \wedge x_n$-t, majd rekurzívan képezzük az $x_1 \wedge \cdots \wedge x_{2k-2} \wedge x_{2k+1} \wedge \cdots \wedge x_n$ konjunkciót minden $1 \le k \le m$-re, és fejezzük be n további lépéssel. Ha $n = 2m - 1$, használjuk ezt a láncot $n + 1$ elemre. Ekkor három lépés törölhető, ha x_{n+1}-nek 1-et adunk értékül. Lásd még Wegener könyvének [63] 3.25. gyakorlatát. Ugyanez az ötlet használható *bármilyen* művelet esetén $\wedge$ helyett, ha az asszociatív és kommutatív.

36. Rekurzívan konstruáljuk meg $P_n(x_1,\ldots,x_n)$-et és $Q_n(x_1,\ldots,x_n)$-et a következőféleképpen, ahol P_n optimális mélységű és Q_n mélysége $\le \lceil \lg n\rceil + 1$: $n=1$ esete triviális, különben pedig P_n-et $Q'_r(x_1,\ldots,x_r)$-ből és $P''_s(x_{r+1},\ldots,x_n)$-ből kapjuk meg, ahol $r=\lceil n/2\rceil$ és $s=\lfloor n/2\rfloor$, még hozzá úgy, hogy legyen $y^j = y'_j$ minden $1\le j\le r$-re és $y^j = y'_r \wedge y''_{j-r}$, ha $r<j\le n$. Q_n-et pedig vagy $P'_r(x_1\wedge x_2,\ldots,x_{n-1}\wedge x_n)$-ből vagy $P'_r(x_1\wedge x_2,\ldots,x_{n-2}\wedge x_{n-1},x_n)$-ből kaphatjuk meg úgy, hogy legyen $y_1=x_1$, $y^{2j}=y'_j$, $y^{2j+1}=y'_j\wedge x^{2j+1}$, ha $1\le j<s$, és legyen $y^{2s}=y'_s$, $y^n=y'_r$.

Az érvényesség bizonyításához azt is meg kell mutatnunk, hogy Q_n y_n kimenetének a mélysége $\lceil \lg n\rceil$. Vegyük észre, hogy Q_{2m+1} hamis lenne, ha $P'_m(x_1\wedge x_2,\ldots,x_{2m-1}\wedge x_{2m})$-mel kezdenénk $P'_{m+1}(x_1\wedge x_2,\ldots,x_{2m-1}\wedge x_{2m},x_{2m+1})$ helyett, *kivéve* ha m 2 hatványa.

Ezek a számítások *minimális memóriával* is elvégezhetőek, egyszerűen legyen $x_{k(i)} \leftarrow x_{j(i)} \wedge x_{k(i)}$ az i. lépésben valamilyen $j(i)<k(i)$ indexekre. Így a konstrukciót ábrázolhatjuk a rendező hálózatoknál is használt diagrammokkal. Például

$P_8 =$ 

; $Q_8 =$ 

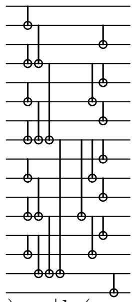
.

A p_n és q_n költségekre teljesülnek a következők: $p_n = \lfloor n/2\rfloor + q_{\lceil n/2\rceil} + p_{\lfloor n/2\rfloor}$ és $q_n = 2\lfloor n/2\rfloor - 1 + p_{\lceil n/2\rceil}$, ha $n>1$. Például $(p_1,\ldots,p_7)=(q_1,\ldots,q_7)=(0,1,2,4,5,7,9)$. Legyen $\bar p_n = 4n-p_n$ és $\bar q_n = 3n - q_n$, ekkor egyszerűbb formulákhoz juthatunk, amik bizonyítják, hogy $p_n<4n$ és $q_n<3n$: $\bar q_n = \bar p_{\lceil n/2\rceil} + [n \text{ páros}]$; $\bar p_{4n} = \bar p_{2n}+\bar p_n + 1$, $\bar p_{4n+1} = \bar p_{2n} + \bar p_{n+1} + 1$, $\bar p_{4n+2} = \bar p_{2n+1} + \bar p_{n+1}$, $\bar p_{4n+3} = \bar p_{4n+2} + 2$. Speciálisan $1+\bar p_{2^m} = F_{m+5}$ egy Fibonacci szám.

Lásd még [29]. Valamivel jobb láncokat lehet készíteni, ha a különben nem megengedett $P'_{\lfloor n/2\rfloor}$ konstrukcióját használjuk Q_n előállításra, ha $n=2^m+1$, ha P_5-öt és P_6-ot Q_5-tel és Q_6-tal helyettesítjük, és ha ekkor felcseréljük $(P_9,P_{10},P_{11},P_{17})$-et $(Q_9,Q_{10},Q_{11},Q_{17})$-tel.

Vegyük észre, hogy ez a konstrukció általában is működik, ha felcseréljük az '$\wedge$'-t *bármely* asszociatív művelettel. Speciálisan az $x_1\oplus\cdots\oplus x_k$ $(1\le k\le n)$ prefixeinek sorozata éppen a konverziót adja meg a Gray-kód és a radix-2 egészek között, lásd még 7.2.1.1–(10).

37. Az $m=15$, $n=16$ eset ábráját lásd jobb oldalt.



(a) Jelölje $x_{i..j}$ az $x_i\wedge\cdots\wedge x_j$ konjunkció eredeti értékét. Meg lehet mutatni, hogy valahányszor az algoritmus elvégzi az $x_k\leftarrow x_j\wedge x_k$ értékadást, x_k előző értéke $x_{j+1..k}$. Az S1 lépés után x_k éppen $x_{f(k)+1..k}$, ahol $f(k)=k\,\&\,(k-1)$, ha $1\le k<m$, és $f(m)=0$. Az S2 lépés után pedig x_k értéke $x_{1..k}$ minden $1\le k\le m$-re.

(b) S1 költsége $m-1$, S2 költsége $m-1-\lceil \lg m\rceil$ és S3-é pedig $n-m$. x_k végső késése $\lfloor \lg k\rfloor + \nu k - 1$, ha $1\le k<m$, és $\lceil \lg m\rceil + k - m$, ha $m\le k\le n$. Így a maximális késés $\{x_1,\ldots,x_{m-1}\}$-re éppen $g(m)=m-1$, ha $m<4$, és $g(m)=\lfloor \lg m\rfloor + \lfloor \lg \frac{m}{3}\rfloor$, ha $m\ge 4$. Így azt kapjuk, hogy $c(m,n)=m+n-2-\lceil \lg m\rceil$ és $d(m,n)=\max(g(m),\lceil \lg m\rceil + n - m)$. Tehát $c(m,n)+d(m,n)=2n-2$, ha $n\ge m+g(m)-\lceil \lg m\rceil$.

(c) Az értékek táblázatát felírva kaphatjuk, hogy $d(n)=\lceil \lg n\rceil$, ha $n<8$, és $d(n)=\lfloor \lg(n-\lfloor \lg n\rfloor + 3)\rfloor + \lfloor \lg \frac{2}{3}(n-\lfloor \lg n\rfloor+3)\rfloor - 1$, ha $n\ge 8$. Máshogy fogalmazva $d(n)>d(n-1)>0$ pontosan akkor teljesül, ha $n=2^k+k-3$ vagy $2^k+2^{k-1}+k-3$ valamely $k>1$ értékre. A minimum $m=n$ mellett vétetik fel, ha $n<8$, különben pedig $m=n-\lfloor \frac{2}{3}(n-\lfloor \lg n\rfloor+3)\rfloor + 2 - [n=2^k+k-3$ valamilyen k-ra$]$ mellett.

(d) Legyen $m\leftarrow m(n,d)$, ahol $m(n,d(n))$ legyen az előző állításban definiált érték, és $m(n,d)=m(n-1,d-1)$, ha $d>d(n)$. Lásd még Snir cikkét [53].

38. (a) Minden $f_k(x_1,\ldots,x_n)$ egy elemi szimmetrikus függvény, az úgynevezett $S_{\ge k}(x_1,\ldots,x_n)$

küszöb függvény. (Lásd még az 5.3.4–28 gyakorlatot és a (7.1.1–90) képletet.)

(b) Miután kiszámítottuk $\{S_1, \ldots, S_n\}$-t $\approx 6n$ lépésben a 33. gyakorlat megoldásának megfelelően, alkalmazhatjuk a 37. gyakorlat módszerét és további $2n$ lépésben végezhetünk is.

De sokkal érdekesebb tervezni egy Boole-láncot külön a $2^m{+}1$ $g_k(x_1, \ldots, x_m) = [(x_1 \ldots x_m)_2 \geq k]$ $(0 \leq k \leq 2^m)$ küszöb függvények kiszámítására. Mivel $[(x'x'')_2 \geq (y'y'')_2] = [(x')_2 \geq (y')_2{+}1] \vee ([(x')_2 \geq (y')_2] \wedge [(x'')_2 \geq (y'')_2])$, egy oszd meg és uralkodj konstrukció megfelel egy bináris dekódolónak, ami ezt a problémát legfeljebb $2t(m)$ költséggel megoldja.

Továbbá, ha $2^{m-1} \leq n < 2^m$, $\{g_1, \ldots, g_n\}$ ezzel a módszerrel való kiszámításának $u(n)$ költsége éppen $2n + O(\sqrt{n}\,)$, és ez egészen elfogadható kis n-ekre:

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$u(n)$ =	0	1	2	4	7	7	8	12	15	17	19	19	20	21	22	27	32	34	36	36

Kezdve a tömörítő összegek kiszámításával, n Boole értéket $s(n) + u(n) \approx 7n$ lépésben rendezhetünk. Egy rendező hálózat $2\hat{S}(n)$ költséggel jobb, ha $n = 4$, de rosszabb, ha $n \geq 8$. Lásd még 5.3.4–(11), és Muller és Preparata cikkét [35].

39. Lásd Paterson cikkét [38]. A következő azonosság azt mutatja, hogy $C(M_{r+s}) \leq C(M_r) + 2^{r+s} + 2^r(2^s - 1) + t(s)$, ahol $t(s)$ a (30) kódoló költsége:

$$M_{r+s}(x_1, \ldots, x_r, x_{r+1}, \ldots, x_{r+s}; y_0, \ldots, y_{2^{r+s}-1}) = M_r(x_1, \ldots, x_r; y'_0, \ldots, y'_{2^r-1}),$$

ahol $y'_j = \bigvee_{k=0}^{2^s-1}(d_k \wedge y_{2^s j+k})$ és d_k a k-adik kimenete egy $(x_{r+1}, \ldots, x_{r+s})$-re alkalmazott s-ből-2^s kódolónak. A mélység $D(M_{r+s}) = \max\bigl(D_x(M_{r+s}), D_y(M_{r+s})\bigr)$, ahol D_x és D_y jelöli az x és y változók maximális mélységét. Ekkor $D_x(M_{r+s}) \leq \max\bigl(D_x(M_r), 1 + s + \lceil \lg s \rceil + D_y(M_r)\bigr)$ és $D_y(M_{r+s}) \leq 1 + s + D_y(M_r)$ teljesül.

Ez $r = \lceil m/2 \rceil$ és $s = \lfloor m/2 \rfloor$ esetén a következő eredményeket adja: $C(M_m) \leq 2^{m+1} + O(2^{m/2})$, $D_y(M_m) \leq m + 1 + \lceil \lg m \rceil$, és $D_x(M_m) \leq D_y(M_m) + \lceil \lg m \rceil$.

40. Legyen például $f_{nk}(x) = \bigvee_{j=1}^{n+1-k}(l_j(x) \wedge r_{j+k-1}(x))$, ahol

$$l_j(x) = \begin{cases} x_j, & \text{ha } j \mod k = 0, \\ x_j \wedge l_{j+1}(x), & \text{ha } j \mod k \neq 0, \end{cases} \quad \text{minden } 1 \leq j \leq n - (n \mod k)\text{-ra;}$$

$$r_j(x) = \begin{cases} 1, & \text{ha } j \mod k = 0, \\ x_j \wedge r_{j-1}(x), & \text{ha } j \mod k \neq 0, \end{cases} \quad \text{minden } k \leq j \leq n\text{-re.}$$

A költség $4n - 3k - 3\lfloor \frac{n}{k} \rfloor - \lfloor \frac{n-1}{k} \rfloor + 2 - (n \mod k)$.

Ha n vagy k kicsi, akkor egy rekurzív megoldás sokkal jobb lenne: Figyeljük meg, hogy

$$f_{nk}(x) = \begin{cases} x_{n-k+1} \wedge \cdots \wedge x_k \wedge \\ \qquad f_{(2n-2k)(n-k)}(x_1, \ldots, x_{n-k}, x_{k+1}, \ldots, x_n), & \text{ha } k < n < 2k; \\ f_{\lfloor (n+k)/2 \rfloor k}(x_1, \ldots, x_{\lfloor (n+k)/2 \rfloor}) \vee \\ \qquad f_{\lfloor (n+k-1)/2 \rfloor k}(x_{\lfloor (n-k)/2 \rfloor + 1}, \ldots, x_n), & \text{ha } n \geq 2k. \end{cases}$$

Ennek a megoldásnak a költsége $n - 1 + \sum_{j=1}^{n-k} \lfloor \lg j \rfloor$, ha $k \leq n < 2k$, és ez aszimptotikusan $(m + \alpha_k - 1)n + O(km)$ és $(m + 2 - 2/\alpha_k)n + O(km)$ között helyezkedik el, ha $n \to \infty$, $m = \lfloor \lg k \rfloor$ és $1 < \alpha_k = (k+1)/2^m \leq 2$.

A két módszer ötvözése még jobb eredményt ad. Az optimális költség nem ismert.

41. Legyen $c(m)$ az $(x)_2 + (y)_2$ és $(x)_2 + (y)_2 + 1$ összegek együttes kiszámításának költsége a conditional sum módszerrel, ha x és y $n = 2^m$ bites, és legyen $c'(m)$ az egyszerűbb probléma, azaz egyedül $(x)_2 + (y)_2$ kiszámításának költsége. Ekkor $c(m+1) = 2c(m) + 6 \cdot 2^m + 2$ és $c'(m+1) =$

$c(m) + c'(m) + 3 \cdot 2^m + 1$. (Az összeg z_n bitjének költsége csak 1, de a z_k biteké ($n < k \leq 2n+1$) már 3, mivel $c?\ a_k$: b_k alakúak, ahol c az átvitel bit.) Ha $n = 1$-gyel, $c(0) = 3$-mal és $c'(0) = 2$-vel indítjuk a rekurziót, akkor a megoldás $c(m) = (3m+5)2^m - 2$, $c'(m) = (3m+2)2^m - m$. De $n = 2$ esetére vannak jobb konstrukciók is, így kezdhetünk $c(1) = 11$-gyel és $c'(1) = 7$-tel is, és ekkor a megoldás $c(m) = (3m + \frac{7}{2})2^m - 2$, $c'(m) = (3m + \frac{1}{2})2^m - m + 1$. Mindkét esetben a mélység $2m + 1$. Lásd még Sklansky cikkét [52].

42. (a) Mivel $\langle x_k y_k c_k \rangle = u_k \vee (v_k \wedge c_k)$, alkalmazhatjuk (26) és az állítás adódik indukcióval.

(b) Vegyük észre, hogy $U_k^{k+1} = u_k$ és $V_k^{k+1} = v_k$. Majd alkalmazzunk indukciót $j - i$ szerint. Lásd még A. Weinberger és Smith cikkét [64], valamint Brent és Kung cikkét [12].

(c) Először is minden $l = 1, 2, \ldots, m-1$ és $1 \leq k \leq n$ értékre számítsuk ki V_i^k-t $h(l)$ minden olyan i többszörösére, amire $k_l \geq i \geq k_{l+1}$ teljesül, ahol $k_l = h(l)\lfloor (k-1)/h(l) \rfloor$ jelölje $h(l)$ legnagyobb k-nál kisebb többszörösét. Például, ha $l = 2$ és $k = 99$, akkor a következőket számítjuk ki: V_{96}^{99}, $V_{88}^{99} = V_{96}^{99} \wedge V_{88}^{96}$, $V_{80}^{99} = V_{88}^{99} \wedge V_{80}^{88}$, $\ldots$, $V_{64}^{99} = V_{72}^{99} \wedge V_{64}^{72}$; ez egy prefix számítás a V_{96}^{99}, V_{88}^{96}, V_{80}^{88}, $\ldots$, V_{64}^{72} értékek felhasználásával, amiket már korábban kiszámítottunk. A 36. gyakorlat módszerével az l. lépés legfeljebb l-lel növeli meg a mélységet és összesen $(p_1 + p_2 + \cdots + p_{2^l})n/2^l = O(2^l n)$ kapura van szükség hozzá.

Majd ismét minden $l = 1, 2, \ldots, m-1$-re és $1 \leq k \leq n$-re számítsuk ki az U_i^k értékeket $i = k_{l+1}$ esetében, felhasználva a következő „kifejtett” formulát:

$$U_{k_{l+1}}^k \;=\; U_{k_l}^k \vee \bigvee_{\substack{k_l > j \geq k_{l+1} \\ h(l)æ}} (V_{j+h(l)}^k \wedge U_j^{j+h(l)}).$$

Például, ha $l = 3$ és $k = 99$, akkor ez a következő formula:

$$U_{64}^{99} \;=\; U_{96}^{99} \vee (V_{96}^{99} \wedge U_{88}^{96}) \vee (V_{88}^{99} \wedge U_{80}^{88}) \vee (V_{80}^{99} \wedge U_{72}^{80}) \vee (V_{72}^{99} \wedge U_{64}^{72}).$$

Minden ilyen U_i^k legfeljebb 2^l term uniója, így legfeljebb l mélység mellett kiszámítható a termek mélységét figyelmen kívül hagyva. A fázis teljes költsége $(0 + 2 + 4 + \cdots + (2^l - 2))n/2^l = O(2^l n)$, ha $1 \leq k \leq n$.

A szükséges U-k és V-k kiszámításának teljes költsége tehát $\sum_{l=1}^{m-1} O(2^l n) = O(2^m n)$. (Mi több a V_0^k értékekre tulajdonképpen nincs is szükség, így $\sum_{l=1}^{m-1} h(l) p_{2^l}$ kapu költsége megtakarítható.) Pédául, ha $m = (2, 3, 4, 5)$, akkor a $(2, 8, 64, 1024)$-bites számok összeadására kapható lánc teljes mélysége rendre $(3, 7, 11, 16)$, illetve teljes költsége $(7, 64, 1254, 48470)$. Ez a konstrukció Khrapchenkonak köszönhető [25], aki azt is megmutatta, hogyan lehet más módszerekkel kombinálva elérni $O(n)$ költséget $\lg n + O(\sqrt{\lg n})$ mélység mellett. Azonban kombinált módszere csak elméleti szempontból érdekes, ugyanis csak $n > 2^{64}$ esetén lesz a mélység kisebb mint $2 \lg n$. Egy másik módja az alacsony mélység elérésének (b) rekurzív kifejezéseivel alapulhat a Fibonacci számokon: A Fibonacci módszer az átviteleket $\lg_\phi n + O(1) \approx 1.44 \lg n$ mélység és $O(n \lg n)$ költség mellett számítja ki. Például a következő tulajdonságú láncokat eredményezi a bináris összeadásra:

n =	4	8	16	32	64	128	256	512	1024
mélység	6	7	9	10	12	13	15	16	18
költség	24	71	186	467	1125	2648	6102	13775	30861

Lásd [26].

Charles Babbage talált egy szellemes mechanikus megoldást a radix 10 számok összeadásának analóg problémájára és azt állította, hogy tetszőleges pontossággal képes elvégezni az összeadást konstans időben. De ehhez persze olyan tökéletes alkatrészekre lett volna szüksége, amelyek nem léteznek. Lásd [5, 334–335. oldal]. Érdekes módon egy egyenértékű ötlet tökéletesen működik tranzisztorok esetében, bár ezt nem lehet kifejezni Boole-láncokkal; lásd Fenwick cikkét [17].

43. (a) Legyen $A = B = Q = \{0, 1\}$ és $q_0 = 0$. Továbbá legyen $c(q, a) = d(q, a) = \bar{q} \wedge a$.

(b) Először konstruáljuk meg a $d_1(q) \ldots d_{n-1}(q)$ függvényeket, ahol $d_1(q) = d(q, a_1)$ és $d_{j+1}(q) = d(d_j(q), a_j)$. Másképp fogalmazva $d_1 = d_{(a_1)}$ és $d_{j+1} = d_j \circ d_{(a_j)}$, ahol $d_{(a)}$ a $q \mapsto d(q,a)$ függvény és $\circ$ a függvénykompozíciót jelöli. Minden d_j függvény bináris alakra kódolható és ekkor $\circ$ egy asszociatív művelet ezeken a bináris reprezentációkon. Így a $d_1 d_2 \ldots d_{n-1}$ függvények a $d^{(a_1)}$, $d_{(a_1)} \circ d^{(a_2)}, \ldots, d^{(a_1)} \circ \cdots \circ d^{(a_{n-1})}$ prefixeknek felelnek meg, és $q_1 q_2 \ldots q_n = q_0 @ d_1(q_0) \ldots d_{n-1}(q_0)$.

(c) Reprezentáljunk egy $f(q)$ függvényt az $f_0 f_1$ igazságtáblájával. Ekkor az $f_0 f_1 \circ g_0 g_1$ kompozíció $h_0 h_1$, ahol a $h_0 = f_0?\ g_1\colon g_0$ és $h_1 = f_1?\ g_1\colon g_0$ mux függvények, melyek mindegyike 3 költséggel és 2 mélységgel kiszámítható. (A kombinált $C(h_0 h_1)$ költség csak 5 lenne, de a mélységet minél alacsonyabban próbáljuk tartani.) $d^{(a)}$ igazságtáblája $a0$. A 36. gyakorlat felhasználásával, a $d_{10} d_{11} d_{20} d_{21} \ldots d_{(n-1)0} d_{(n-1)1}$ igazságtáblákat legfeljebb $6p_{n-1} < 24n$ költséggel és legfeljebb $2\lceil \lg(n-1) \rceil$ mélységgel kiszámíthatjuk; továbbá $b_j = \bar{q}_j \wedge a_j = \bar{d}_{(j-1)0} \wedge a_j$. (Ezek a költségbecslések meglehetősen konzervatívak; alapos egyszerűsítésekre lenne lehetőség a kezdeti igazságtáblákban előforduló sok 0 miatt és mert sok köztes d_{j1} érték sosem kerül felhasználásra. Például, ha $n = 5$, a tényleges költség csak 10, nem pedig $6p_4 + 4 = 28$; a tényleges mélység pedig 4, és nem pedig $1 + 2\lceil \lg 4 \rceil = 5$.)

44. A bemenet felfogható egy $x_0 y_0\, x_1 y_1 \ldots x_{n-1} y_{n-1}$ füzérként, aminek az elemei a négyelemű $A = \{00, 01, 10, 11\}$ ábécéből valók; a két állapot $Q = \{0, 1\}$ a lehetséges átvitel bitet reprezentálja, $q_0 = 0$; a kimenet ábécéje pedig $B = \{0, 1\}$; továbbá $c(q, xy) = q \oplus x \oplus y$, $d(q, xy) = \langle qxy \rangle$. Ebben az esetben a véges állapotú átalakító tulajdonképpen leírható egy teljes összeadóval.

A $d_{(xy)}$ leképezések előállításakor q-nak csak három függvénye fordul elő a lehetséges négyből. Ezek lekódolhatóak $u \vee (q \wedge v)$ alakban. A kezdő $d^{(xy)}$ függvényekre $u = x \wedge y$, $v = x \oplus y$ adódik, az $(uv) \circ (u'v')$ kompozícióra pedig $u''v''$, ahol $u'' = u' \vee (v' \wedge u)$ és $v'' = v \wedge v'$.

Ha $n = 4$ például, a lánc a következő formát ölti a 42. gyakorlat jelöléseivel: $U_k^{k+1} = x_k \wedge y_k$, $V_k^{k+1} = x_k \oplus y_k$, ha $0 \le k < 4$; $U_0^2 = U_1^2 \vee (V_1^2 \wedge U_0^1)$, $U_2^4 = U_3^4 \vee (V_3^4 \wedge U_2^3)$, $V_2^4 = V_2^3 \wedge V_3^4$; $U_0^3 = U_2^3 \vee (V_2^3 \wedge U_0^2)$, $U_0^4 = U_2^4 \vee (V_2^4 \wedge U_0^2)$; $z_0 = V_0^1$, $z_1 = U_0^1 \oplus V_1^2$, $z_2 = U_0^2 \oplus V_2^3$, $z_3 = U_0^3 \oplus V_3^4$, $z_4 = U_0^4$. A teljes költség 20 és a maximális mélység 5.

Általános esetben a költségre $2n + 3p_n$ adódik a 36. gyakorlat jelöléseivel, ugyanis $2n$ kapura van szükség a kezdő u-khoz és v-khez, majd $3p_n$ kapura a prefix számításhoz; a további $n-1$ kaput pedig, ami a z_j-k $(0 < j < n)$ kiszámításához szükséges, nem kell figyelembe vennünk, ugyanis a költségekbe már amúgy is beleszámított V_0^j-k kiszámítása szükségtelen $(1 < j \le n)$. Így a teljes költség $14 \cdot 2^m - 3F_{m+5} + 3$, ami nyilvánvalóan felülmúlja a conditional sum módszerrel kapható megoldást (aminek a mélysége szintén $2m + 1$):

$n =$	2	4	8	16	32	64	128	256	512	1024
conditional sum lánc költsége	7	25	74	197	492	1179	2746	6265	14072	31223
Ladner–Fischer-lánc költsége	7	20	52	125	286	632	1363	2888	6040	12509

George Boole azért találta ki az algebráját, hogy megmutassa, a logika aritmetikai kifejezésekkel is leírható. Azóta a helyzet megfordult. Olyanok mint Shannon és Zuse az 1930-as években elkezdtek áramköröket tervezni logikai alapokon aritmetikai számításokra és azóta is számos új megközelítést fedeztek fel a párhuzamos összeadás problémáját vizsgálva. Az első $O(n)$ költségű és $O(\lg n)$ mélységű Boole-láncokat Ofman szerkesztette meg [37]. Az ő láncai hasonlóak voltak a fenti konstrukcióhoz, de a mélységük hozzávetőlegesen $4m$ volt.

45. Ez a fajta érvelés valóban egyszerűbb lenne, de nem elég erős a kitűzött eredmény bizonyításához. (Számos 0 kimenet értékű lépéseket is tartalmazó lánc mutatható, ami felduzzasztja az egyszerűbb becslést.) A fejezet permutációkon alapuló bizonyítási technikáját először Savage alkalmazta könyvében, [47, 3.4.1. tétel].

46. Legyen $r = 2^n/n + O(1)$, ekkor $\ln(2^{2r+1}(n+r-1)^{2r}/(r-1)!) = r \ln r + (1 + \ln 4)r + O(n) = (2^n/n)(n \ln 2 - \ln n + 1 + \ln 4) + O(n)$. Így $\alpha(n) \le (n/(4e))^{-2^n/n + O(n/\lg n)}$, ami csakugyan meglehetősen gyorsan tart nullához, ha $n > 4e$. (És valóban, (32) szerint $\alpha(11) < 7.6 \times 10^7$, $\alpha(12) < 4.2 \times 10^{-6}$,

$\alpha(13) < 1.2 \times 10^{-38}$.)

47. Tekintsük most a permutációknak csak azon $(r-m)!$ esetét, amikor $i\pi = i$ minden $1 \le i \le n$ esetén és $(n{+}r{+}1{-}k)\pi$ a k-adik kimenet. Ekkor (32) helyett $(r-m)!\, c(m,n,r) \le 2^{2r+1}(n{+}r{-}1)^{2r}$ adódik. Ebből pedig a 46. gyakorlat alapján következik, hogy majdnem minden ilyen függvény költsége meghaladja a $2^n m/(n+\lg m)$ értéket, ha $m = O(2^n/n^2)$.

48. (a) Nem meglepő módon a $C(n)$ értékére adott alsó korlát kis n-ekre meglehetősen pontatlan:

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$r(n)$ =	1	1	2	3	5	9	16	29	54	99	184	343	639	1196	2246	4229

(b) A bootstrap módszer (lásd [26, S9.4.]) a következő eredményt adja:

$$r(n) = \frac{2^n}{n}\Big(1 + \frac{\lg n - 2 - 1/\ln 2}{n} + O\Big(\frac{\lg n}{n^2}\Big)\Big).$$

49. Azon normalizált Boole-függvények száma, amelyek reprezentálhatóak r-nél nem hosszabb formulával, legfeljebb $5^r n^{r+1} g_r$, ahol g_r az r belső csúccsal rendelkező irányított bináris fák száma. Helyettesítsünk most r helyébe $2^n/\lg n - 2^{n+2}/(\lg n)^2$-t a képletben és osszunk 2^{2^n-1}-gyel, hogy felső korláthoz jussunk a legfeljebb r hosszú függvények hányadrészére. Az eredmény gyorsan konvergál nullához, lásd a 2.3.4.4–7. gyakorlatot, ugyanis ez $O((5\alpha/16)^{2^n/\lg n})$, ahol $\alpha \approx 2.483$.

Riordan és Shannon egy hasonló alsó korláthoz jutottak soros-párhuzamos kapcsoló hálózatokra [46], az ilyen hálózatok kölcsönösen megfeleltethetőek olyan formuláknak, amelyek csak irányító operátorokat használnak. Krichevsky általánosabb eredményeket is levezetett [28], és Lupanov adott egy aszimptotikusan megegyező felső korlátot [32].

50. (a) Felhasználva a 7.1.1-**??**-as gyakorlatban is alkalmazott alkocka jelölést a prím implikánsokra a következők adódnak: 00001∗, (0001∗1), 0100∗1, 0111∗1, 1010∗1, 101∗11, 00∗011, 00∗101, (01∗111), 11∗101, (0∗1101), (1∗0101), 1∗1011, 0∗0∗11, ∗00101, (∗01011), (∗11101), ahol a zárójelben lévő alkockák elhagyásával adódik a legrövidebb DNF. (b) Hasonlóképpen adódnak a prím klózok és a legrövidebb KNF: 00111∗, 01010∗, 10110∗, 0110∗∗, 00∗00∗, 11∗00∗, 11∗11∗, (0∗100∗), (1∗00∗∗), 1∗0∗1∗, (1∗∗∗∗0), ∗0000∗, (∗1100∗), ∗1∗∗∗0, ∗∗1∗∗0, ∗∗∗1∗0, és (∗∗∗∗00). (Így tehát a KNF $(x_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (x_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4 \vee x_5) \wedge \cdots \wedge (\bar{x}_4 \vee x_6)$.)

51. $f = \big([x_5x_6 \in \{01\}] \wedge [(x_1x_2x_3x_4)_2 \in \{1,3,4,7,9,10,13,15\}]\big) \vee \big([x_5x_6 \in \{10,11\}] \wedge [x_1x_2x_3x_4 = 0000]\big) \vee \big([x_5x_6 \in \{11\}] \wedge [(x_1x_2x_3x_4)_2 \in \{1,2,4,5,7,10,11,14\}]\big)$.

52. A kis n-ekre kapható eredmények jelentősen eltérnek az aszimptotikus értékektől:

n	k	l	(38)	n	k	l	(38)	n	k	l	(38)	n	k	l	(38)
5	2	2	39	8	3	2	175	11	4	4	803	14	5	5	4045
6	2	2	67	9	3	2	279	12	4	3	1329	15	5	5	7141
7	2	1	109	10	4	4	471	13	5	6	2355	16	5	4	12431

(A költségek általában tovább csökkenthetőek olyan optimalizálások révén, mint $[x_1x_2 \in \{00,01\}] = \bar{x}_1$.)

53. Először vegyük észre, hogy $2^k/l \le n - 3\lg n$, így $m_i \le n - 3\lg n + 1$ és $2^{m_i} = O(2^n/n^3)$. Továbbá $l = O(n)$ és $t(n-k) = O(2^n/n^2)$. Ezért (38) a következő egyszerű alakban felírható: $l \cdot 2^{n-k} + O(2^n/n^2) = 2^n/(n - 3\lg n) + O(2^n/n^2)$.

54. A mohó-lábnyom heurisztika a következő 15 hosszú láncot adja:

$$\begin{array}{lll} x_5 = x_1 \oplus x_2, & x_{10} = x_4 \oplus x_5, & f_2 = x_{15} = \bar{x}_5 \wedge x_8, \\ x_6 = x_2 \oplus x_3, & x_{11} = x_7 \vee x_{10}, & f_3 = x_{16} = x_4 \wedge \bar{x}_{12}, \\ x_7 = x_1 \wedge \bar{x}_3, & x_{12} = x_6 \oplus x_{11}, & f_4 = x_{17} = x_1 \wedge x_8, \\ x_8 = x_4 \wedge \bar{x}_6, & x_{13} = \bar{x}_{10} \wedge x_{12}, & f_5 = x_{18} = x_7 \wedge x_9, \\ x_9 = x_4 \wedge \bar{x}_5, & f_1 = x_{14} = x_6 \wedge \bar{x}_{11}, & f_6 = x_{19} = \bar{x}_3 \wedge x_{13}. \end{array}$$

A minterm-first módszer alkalmazása egy 22 hosszú láncot eredményez, miután eltávolítjuk a sohasem használt lépéseket:

$$\begin{array}{lll} x_5 = \bar{x}_1 \wedge \bar{x}_2, & x_{13} = x_5 \wedge x_{10}, & x_{20} = x_8 \wedge x_{11}, \\ x_6 = \bar{x}_1 \wedge x_2, & x_{14} = x_5 \wedge x_{11}, & f_6 = x_{21} = x_{15} \vee x_{18}, \\ x_7 = x_1 \wedge \bar{x}_2, & x_{15} = x_6 \wedge x_9, & f_1 = x_{22} = x_{13} \vee x_{21}, \\ x_8 = x_1 \wedge x_2, & x_{16} = x_6 \wedge x_{11}, & f_2 = x_{23} = x_{12} \vee x_{20}, \\ x_9 = \bar{x}_3 \wedge x_4, & x_{17} = x_7 \wedge x_9, & x_{24} = x_{14} \vee x_{16}, \\ x_{10} = x_3 \wedge \bar{x}_4, & x_{18} = x_7 \wedge x_{11}, & f_3 = x_{25} = x_{24} \vee x_{19}, \\ x_{11} = x_3 \wedge x_4, & f_5 = x_{19} = x_8 \wedge x_9, & f_4 = x_{26} = x_{17} \vee x_{20}. \\ x_{12} = x_5 \wedge x_9, & & \end{array}$$

(A disztributivitási szabály segítségével x_{14}, x_{16} és x_{24} kiszámítását két lépéssel helyettesíthetnénk.)

Egyébként az 51. gyakorlat megoldásában szereplő három függvény csupán tíz lépésben kiszámítható:

$$\begin{array}{lll} x_5 = x_2 \vee x_4, & f_3 = x_9 = x_6 \oplus x_8, & x_{12} = x_2 \oplus x_3, \\ x_6 = \bar{x}_1 \wedge x_5, & x_{10} = x_1 \oplus x_8, & x_{13} = \bar{x}_{10} \wedge x_{12}, \\ x_7 = x_2 \wedge x_4, & \bar{f}_2 = x_{11} = x_9 \vee x_{10}, & f_1 = x_{14} = x_4 \oplus x_{13}. \\ x_8 = x_3 \wedge \bar{x}_7, & & \end{array}$$

55. Az 50. gyakorlat megoldásában kapott optimális kétszintű DNF és KNF reprezentációk költsége 53, illetve 43. A (37) formula költsége 30, ha az 54. gyakorlatnak megfelelően optimalizáljuk. Az 51. gyakorlatban szereplő alternatíva költsége csak 17. Az optimális ötváltozós láncok katalógusa viszont a következő megoldást javasolja erre a hatváltozós függvényre:

$$\begin{array}{llll} x_7 = \bar{x}_1 \wedge x_2, & x_{11} = x_5 \wedge x_{10}, & x_{15} = x_{13} \oplus x_{14}, & x_{18} = \bar{x}_4 \wedge x_{17}, \\ x_8 = x_3 \oplus x_7, & x_{12} = x_5 \vee x_{10}, & x_{16} = x_5 \wedge \bar{x}_{10}, & x_{19} = x_6 \wedge x_{15}, \\ x_9 = x_2 \wedge x_8, & x_{13} = x_4 \wedge \bar{x}_{11}, & x_{17} = \bar{x}_3 \wedge x_{16}, & x_{20} = x_{18} \vee x_{19}, \\ x_{10} = x_1 \oplus x_9, & x_{14} = x_7 \wedge x_{12}, & & \end{array}$$

Van-e vajon jobb megoldás?

56. Ha legfeljebb két érték meghatározott, akkor a függvény konstans vagy x_j vagy pedig $\bar{x}_j$.

57. Az $x_5 \ldots x_{17}$ formulák igazságtáblái hexadecimális jelöléssel rendre `0ff0`, `2222`, `33cc`, `0d0d`, `7777`, `5d5d`, `3ec1`, `6b94`, `4914`, `4804`, `060b`, `2020`, `7007`. Így a következő ”számjegyeket” kapjuk:

$$1010 = 0\,, \quad 1011 = 1\,, \quad 1012 = 2\,, \quad 1013 = 3\,, \quad 1014 = 4\,, \quad 1015 = 5\,.$$

58. Azon 7 költségű függvények igazságtáblái, amelyek igazságtáblái pontosan nyolc egyest tartalmaznak, mind ekvivalensek a következők valamelyikével: `0779`, `169b` vagy `179a`. Ezek valamennyi lehetséges kombinációját előállítva 9656 olyan megoldáshoz juthatunk, amelyek különböznek egymástól figyelembe véve $\{x_1, x_2, x_3, x_4\}$, illetve $\{f_1, f_2, f_3, f_4\}$ esetleges permutációit és komplementereit.

59. A mohó-lábnyom heurisztika a következő 17 lépéses láncot adja:

$$\begin{array}{rlrlrl}
x_5 &= x_1 \vee x_4, & x_{11} &= x_8 \vee x_9, & x_{17} &= \bar{x}_2 \wedge x_3,\\
x_6 &= x_1 \oplus x_3, & x_{12} &= x_1 \oplus x_{11}, & f_1 = x_{18} &= x_{13} \oplus x_{15},\\
x_7 &= x_2 \oplus x_4, & x_{13} &= x_5 \wedge \bar{x}_9, & f_2 = x_{19} &= x_{11} \wedge \bar{x}_{16},\\
x_8 &= \bar{x}_4 \wedge x_6, & x_{14} &= x_5 \wedge x_{12}, & f_3 = x_{20} &= x_{12} \oplus x_{17},\\
x_9 &= x_3 \oplus x_7, & x_{15} &= x_2 \wedge x_6, & f_4 = x_{21} &= x_{10} \wedge \bar{x}_{14}.\\
x_{10} &= x_2 \vee x_3, & x_{16} &= x_2 \wedge \bar{x}_6, & &
\end{array}$$

A kiinduló függvények lábnyoma nagy, így nem érhetjük el, hogy $C(f_1f_2f_3f_4) = 28$ teljesüljön, de egy alig valamivel bonyolultabb S-doboz valószínűleg létezik.

60. Az egyik lehetőség, hogy $u_1 = x_1 \oplus y_1$, $u_2 = x_2 \oplus y_2$, $v_1 = y_2 \oplus u_1$, $v_2 = y_1 \oplus u_2$, $z_1 = v_1 \wedge \bar{u}_2$, $z_2 = v_2 \wedge \bar{u}_1$.

61. Tekintsünk ezekre a hatváltozós parciális függvényekre mint 4×16-os igazságtáblákra, amelyek sorait x_1y_1 határozzák meg. Ekkor a 4-bites függvényekről szerzett ismereteinkkel hatékonyan kiszámíthatjuk a sorokat és ez végül a következő 25 lépésből álló megoldáshoz vezet: $t_1 = x_2 \wedge y_2$, $t_2 = x_3 \wedge y_3$, $t_3 = x_2 \vee y_2$, $t_4 = x_3 \vee y_3$, $t_5 = t_1 \oplus t_2$, $t_6 = t_1 \vee t_2$, $t_7 = t_4 \wedge \bar{t}_5$, $t_8 = t_3 \oplus t_6$, $t_9 = x_2 \oplus y_2$, $t_{10} = t_4 \oplus t_9$, $t_{11} = t_5 \wedge \bar{t}_{10}$, $t_{12} = t_3 \oplus t_4$, $t_{13} = x_1 \vee y_1$, $t_{14} = t_8 \oplus t_{12}$, $t_{15} = t_{13} \wedge \bar{t}_{14}$, $t_{16} = t_4 \oplus t_7$, $t_{17} = t_{13} \wedge \bar{t}_{16}$, $t_{18} = t_3 \vee t_4$, $t_{19} = x_1 \oplus y_1$, $t_{20} = t_{19} \wedge \bar{t}_{18}$, $t_{21} = t_8 \oplus t_{15}$, $t_{22} = t_7 \oplus t_{17}$, $z_1 = t_{11} \vee t_{20}$, $z_2 = t_{21} \wedge \bar{t}_{20}$, $z_3 = t_{22} \wedge \bar{t}_{20}$. (Létezik-e jobb megoldás? Gilbert Lee talált egy 17 lépéses megoldást arra az esetre, amikor a bemenet csak 000, 001, 011, 101 vagy 111 lehet.)

62. Összesen $\binom{2^n}{2^n d}2^{2^n c}$ ilyen függvény létezik és legfeljebb $\binom{2^n}{2^n d}t(n,r)$ költsége $\le r$. Így a 46. gyakorlatban is alkalmazott érveléssel megmutathatjuk (32) felhasználásával, hogy a legfeljebb $r = \lfloor 2^n c/n \rfloor$ költségű függvények hányadrésze legfeljebb $2^{2r+1-2^n c}(n+r-1)^{2r}/(r-1)! = 2^{-r \lg n + O(r)}$.

63. [51] Írjuk fel az igazságtáblát egy $2^k \times 2^{n-k}$ méretű táblázatba, ahogyan Lupanov módszerében is tettük, és tegyük fel, hogy a j. oszlopban c_j érték van megadva – a többi don't-care – minden $0 \le j < 2^{n-k}$-ra. Majd osszuk fel az oszlopot $\lfloor c_j/m \rfloor + 1$ részre úgy, hogy a legalsó esetlegesen üres részoszlop kivételével minden rész m definiált értéket tartalmazzon. A segítség szerint legfeljebb 2^{m+k} oszlopvektor elegendő az összes 0 és 1 lefedéséhez azokban a részoszlopokban, amelyek legfelső és legalsó sora egy-egy rögzített i_0, illetve i_1 indexhez tartozik. Ekkor $O(m2^{m+3k})$ művelettel megkonstruálhatunk $O(2^{m+3k})$ $g_t(x_1, \ldots, x_k)$ függvényt az $\{x_1, \ldots, x_k\}$ mintermjeiből úgy, hogy minden részoszlop illeszkedjen egy t típusra. És minden t típusra $\{x_{k+1}, \ldots, x_n\}$ mintermejeiből megszerkeszthetünk olyan $h_t(x_{k+1}, \ldots, x_n)$ függvényeket, amik megadják, hogy melyik oszlopok illeszkednek egy adott t-re. A költség legfeljebb $\sum_j(\lfloor c_j/m \rfloor + 1) \le 2^n c/m + 2^{n-k}$. Végül pedig $f = \bigvee_t(g_t \wedge h_t)$ további $O(2^{m+3k})$ lépést igényel. A teljes költségre tehát legfeljebb $(2^n c/n)(1 + 9n\_ \lg n + O(n\_))$, ha a $k = \lfloor 2 \lg n \rfloor$, $m = \lceil n - 9 \lg n \rceil$ értékeket választjuk.

Természetesen a segítségben megfogalmazott állítást is be kellene bizonyítani, ami Necsiporuknak köszönhető [36]. Voltaképpen $2^m(1 + \lceil k \ln 2 \rceil)$ vektor is elegendő (lásd [55]): Ha véletlenszerűen választunk $q = 2^m \lceil k \ln 2 \rceil$ nem feltétlenül különböző vektort, akkor az érintetlenül hagyott részkockák várható száma $\binom{k}{m}2^m(1-2^{-m})^q < \binom{k}{m}2^m e^{-q2^{-m}} < 2^m$. (Egy explicit bizonyításnak persze jobban örülnénk.)

További széles körű általánosítások – bizonyos mértékű hibatolerancia kezelése és az egyesek sűrűségének megadása – megtalálhatóak Pippenger cikkében [40].

64. Ez éppen az amőba, ha a mezőket így számozzuk meg:
618753294 - éppen ahogy egy ősi kínai bűvös négyzeten. Berlekamp, Conway, és Guy ezt a számozási sémát használják az amőba teljes elemzésekor könyvükben [8].

65. Az egyik megoldás, hogy a „védekező” d_j lépéseket felcseréljük „támadó” a_j és „ellentámadó” c_j lépésekre, és csak a $j \in \{1, 3, 9, 7\}$ sarok mezőkben alkalmazzuk őket. Legyen $j \cdot k = (jk)$ mod 10, ekkor

$$\begin{array}{ccc} j\cdot 1 & j\cdot 2 & j\cdot 3 \\ j\cdot 4 & j\cdot 5 & j\cdot 6 \\ j\cdot 7 & j\cdot 8 & j\cdot 9 \end{array}$$

egy újabb ábrázolása az amőbának, ha j egy sarok mező, ugyanis $j \perp 10$. Így a_j és c_j pontos definíciója:

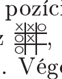
$$\begin{aligned} a_j &= m_j \wedge \big((x_{j\cdot 3} \wedge \beta_{(j\cdot 8)(j\cdot 9)} \wedge (o_{j\cdot 4} \oplus o_{j\cdot 6})\big) \vee \big(x_{j\cdot 7} \wedge \beta_{(j\cdot 6)(j\cdot 9)} \wedge (o_{j\cdot 2} \oplus o_{j\cdot 8})\big) \\ &\qquad\qquad \vee \big(m_{j\cdot 9} \wedge \big((m_{j\cdot 8} \wedge x_{j\cdot 2} \wedge \overline{(o_{j\cdot 3} \oplus o_{j\cdot 6})}) \vee (m_{j\cdot 6} \wedge x_{j\cdot 4} \wedge \overline{(o_{j\cdot 7} \oplus o_{j\cdot 8})})\big)\big)\big); \\ c_j &= d_j \wedge \overline{(x_{j\cdot 6} \wedge o_{j\cdot 7})} \wedge \overline{(x_{j\cdot 8} \wedge o_{j\cdot 3})} \wedge \bar{d}_{j\cdot 9}; \end{aligned}$$

itt $d_j = m_j \wedge \beta_{(j\cdot 2)(j\cdot 3)} \wedge \beta_{(j\cdot 4)(j\cdot 7)}$ helyettesíti (51)-et. Továbbá legyen

$$\begin{aligned} u &= (x_1 \oplus x_3) \oplus (x_7 \oplus x_9), \\ v &= (o_1 \oplus o_3) \oplus (o_7 \oplus o_9), \\ t &= m_2 \wedge m_6 \wedge m_8 \wedge m_4 \wedge (u \vee \bar{v}), \end{aligned} \qquad z_j = \begin{cases} m_j \wedge \bar{t}, & \text{ha } j = 5, \\ m_j \wedge \bar{d}_{j\cdot 9}, & \text{ha } j \in \{1, 3, 9, 7\}, \\ m_j, & \text{ha } j \in \{2, 6, 8, 4\}, \end{cases}$$

hogy kezeljünk még néhány kivételes esetet. Végül a lépések rangsorolt sorozatát, $d_5d_1d_3d_9d_7d_2d_6$ $d_8d_4m_5m_1m_3m_9m_7m_2m_6m_8m_4$-et (53)-ban helyettesítsük $a_1a_3a_9a_7c_1c_3c_9c_7z_5z_1z_3z_9z_7z_2z_6z_8z_4$-gyel, és cseréljük fel $(d_j \wedge \bar{d}'_j) \vee (m_j \wedge \bar{m}'_j)$-t (55)-ben $(a_j \wedge \bar{a}'_j) \vee (c_j \wedge \bar{c}'_j) \vee (z_j \wedge \bar{z}'_j)$-vel, ha j egy sarok mező, különben pedig egyszerűen $(z_j \wedge \bar{z}'_j)$-vel.

(Megjegyezzük, hogy ennek a gépnek *valamennyi* megengedett pozícióból jól kell tudnia lépnie, még olyan pozíciókból is, amelyek nem fordulhatnak elő a gép korábbi lépései miatt. Felfogható úgy is, hogy megengedjük az emberi játékosnak, hogy egy ideig maga játsszon és csak utána hívja segítségül a gépet. Ellenkező esetben jelentős egyszerűsítésekre lenne lehetőség. Például, ha mindig $\times$ kezd, akkor első lépésként elfoglalhatja a középső mezőt, ezzel kiiktatva számos jövőbeli lehetőséget. Ekkor kevesebb mint $8 \times 6 \times 4 \times 2 = 384$ játszma fordulhatna elő. Még akkor is, ha $\emptyset/$ kezd, kevesebb mint $9 \times 7 \times 5 \times 3 = 945$ lehetséges forgatókönyv létezik egy rögzített stratégia mellett. Tulajdonképpen a különböző játszmák valódi száma az itt definiált stratégia mellett $76 + 457$, amiből $72 + 328$-at a gép nyer meg, míg a maradék pedig döntetlen.)

66. Az előző megoldásban látott Boole-lánc mind a 4520 megengedett pozícióból helyes lépést adott, ahol a helyesség azt jelentette, hogy az adott pozícióból elérhető legrosszabb befejezést maximalizáltuk. De egy igazán jó amőba játékos máshogy csinálná. Például a  pozícióból a gép a középső mezőt választja, , és ezután O valószínűleg az egyik sarkot, ezzel döntetlent elérve. Azonban a  pozíciót választva O már csak két lépés közül választhatna, ha el akarja kerülni a vereséget. Lásd Martin Gardner könyvét [19, 4. fejezet]. Továbbá az olyan pozícióból mint amilyen , a legjobb lépés  az azonnali győzelem helyett, ekkor, ha a válasz , lépjünk -et. Ily módon még mindig győzünk, de nem alázzuk meg annyira ellenfelünket. Végezetül, az egyetlen „legjobb lépésnek” már a koncepciója is sántít, ugyanis egy jó játékos különböző lépéseket fog választani különböző játszmákban (ahogy ezt Babbage is megfigyelte).

Azt gondolnánk, hogy egyszerű egy digitális számítógépet beprogramozni

amőbázni vagy speciális áramköröket tervezni egy amőbagéphez.
Ez igaz, hacsak nem egy mester robotot akarunk létrehozni,
ami a lehető legtöbb játékot nyeri meg tapasztalatlan játékosok ellen.

– MARTIN GARDNER, *The Scientific American Book of Mathematical Puzzles & Diversions* (1959) [20]

67. A szerző által elért legjobb eredmény 1734 kaput használ fel és Sholomov 63. gyakorlat megoldásában látott módszerének alkalmazásával sikerült elérni: Először osszuk fel az igazságtáblákat 64 sorra $o_5x_5o_2o_6o_8o_4$ szerint és 4096 oszlopra a többi 12 bemeneti változónak megfelelően. Majd írjunk egyeseket a „care” helyekre úgy, hogy az oszlopokban viszonylag kevés egyes legyen. Ezután keressünk néhány oszloptípust, amelyek illeszkednek valamennyi oszlopban a definiált helyekre. 23 típus elegendő a c függvényhez, 20 típus az s-hez és 6 az m-hez. Ekkor valamennyi kimenetet kiszámíthatjuk $\bigvee(g_t \wedge h_t)$ alakban, ezáltal a szükséges minterm számítások jelentős részét szétosztva g_t-ben és h_t-ban. Ezt a feladatot Wakerly könyvének egyik kérdése ihlette, [62, S6.2.7.] Egyébként Babbage úgy akart választani k lehetséges lépés közül, hogy megvizsgálta $N \mod k$-t, ahol N az eddig megnyert játszmák száma volt. Nem ismerte fel, hogy az egymást követő lépések N megváltozásáig erősen összefüggnek. Sokkal jobb választás lett volna N-nek az eddig *megtett lépések* száma.

68. Nem. Az a módszer egy „egységes” láncot eredményez jól átlátható szerkezettel, de a költsége 2^n-szer egy n-ben polinomiális kifejezés. Az L tétel segítségével megkonstruálható egy áramkör hozzávetőlegesen $2^n/n$ kapuval, de sokkal nehezebb elkészíteni. (Egyébként $C(\pi_5) = 10$.)

69. (a) Például le lehet ellenőrizni az eredményt mind a 64 eset megvizsgálásával.

(b) Ha x_m ugyanabban a sorban vagy oszlopban van mint x_i és szintén ugyanabban a sorban vagy oszlopban mint x_j, akkor $\alpha_{111} = \alpha_{101} = \alpha_{110} = 0$, tehát a párok jók. Ellenkező esetben lényegében véve három különböző lehetőség van és mind rossz: Ha $(i,j,m) = (1,2,4)$, akkor $\alpha_{101} = 0$, $\alpha_{100} = x_5x_9 \oplus x_6x_8$, $\alpha_{011} = x_9$; ha $(i,j,m) = (1,2,6)$, akkor $\alpha_{010} = x_4x_9$, $\alpha_{011} = x_7$, $\alpha_{100} = x_5x_9$, $\alpha_{101} = x_8$; ha $(i,j,m) = (1,5,9)$, akkor $\alpha_{111} = 1$, $\alpha_{110} = 0$, $\alpha_{010} = x_3x_7$.

70. (a) $x_1\wedge((x_5\wedge x_9)\oplus(x_6\wedge x_8)) \oplus x_2\wedge((x_6\wedge x_7)\oplus(x_4\wedge x_9)) \oplus x_3\wedge((x_4\wedge x_8)\oplus(x_5\wedge x_7))$.

(b) $x_1\wedge((x_5\wedge x_9)\vee(x_6\wedge x_8)) \vee x_2\wedge((x_6\wedge x_7)\vee(x_4\wedge x_9)) \vee x_3\wedge((x_4\wedge x_8)\vee(x_5\wedge x_7))$.

(c) Legyen $y_1 = x_1 \wedge x_5 \wedge x_9$, $y_2 = x_1 \wedge x_6 \wedge x_8$, $y_3 = x_2 \wedge x_6 \wedge x_7$, $y_4 = x_2 \wedge x_4 \wedge x_9$, $y_5 = x_3 \wedge x_4 \wedge x_8$, $y_6 = x_3 \wedge x_5 \wedge x_7$. Az $f(y_1,\ldots,y_6) = [y_1 + y_2 + y_3 > y_4 + y_5 + y_6]$ függvény további 15 lépésben kiértékelhető két teljes összeadó és egy összehasonlító segítségével, de létezik egy 14 lépéses megoldás is: Legyen $z_1 = (y_1\oplus y_2)\oplus y_3$, $z_2 = (y_1\oplus y_2)\vee(y_1\oplus y_3)$, $z_3 = (y_4\oplus y_5)\oplus y_6$, $z_4 = (y_4 \oplus y_5) \vee (y_4 \oplus y_6)$. Ekkor $f = (z_1 \oplus (z_2\wedge(\bar z_4\oplus(z_1\vee z_3)))) \wedge (\bar z_3\vee z_4)$. Továbbá $y_1y_2y_3 = 111 \iff y_4y_5y_6 = 111$, tehát vannak don't-care helyek, ami egy 11 lépéses megoldáshoz vezet: $f = ((\bar z_1\wedge z_3)\vee\bar z_4) \wedge z_2$. Így a teljes költség $12 + 11 = 23$.

(A szerző nem ismer módszert arra, hogy a számítógép hogyan találhatna meg egy ennyire hatékony láncot belátható időn belül, ha f-nek csak az igazságtáblája adott. De talán még ennél is létezik hatékonyabb lánc.)

71. (a) $P(p) = 1-12p^2+24p^3+12p^4-96p^5+144p^6-96p^7+24p^8$, ami $\frac{11}{32}+\frac{9}{2}\epsilon^2-3\epsilon^4-24\epsilon^6+24\epsilon^8$, ha $p = \frac{1}{2} + \epsilon$.

(b) Az $(f_0,\ldots,f_7)$ nyolcasokat összesen $N = 2^{n-3}$ módon lehet halmazokba sorolni, melyek mindegyike $P(p)$ valószínűséggel ad jó párokat. Így tehát a válasz $1 - P(p)^N$.

(c) Annak a valószínűsége, hogy pontosan r halmaz jó, $\binom{N}{r}P(p)^r(1-P(p))^{N-r}$. Ekkor t próba $(r/N)^t$ valószínűséggel talál jó párokat. Így a válasz $1 - \sum_{r=0}^{N}\binom{N}{r}P(p)^r(1-P(p))^{N-r}(r/N)^t = 1 - P(p)^t + O(t^2/N)$.

(d) $\sum_{r=0}^{N}\binom{N}{r}P(p)^r(1-P(p))^{N-r}\sum_{j=0}^{t-1}(r/N)^j = (1-P(p)^t)/(1-P(p)) + O(t^3/N)$.

72. A 71(a) gyakorlatra kapott valószínűség a következőképpen módosul: $P(p) + (72p^3 - 264p^4 + 432p^5 - 336p^6 + 96p^7)r + (60p^2 - 240p^3 + 456p^4 - 432p^5 + 144p^6)r^2 + (-48p^2 + 144p^3 - 216p^4 + 96p^5)r^3 + (-36p^2 + 24p^3 + 12p^4)r^4 + (48p^2 - 24p^3)r^5 - 12p^2r^6$. Ha $p = q = (1-r)/2$, akkor ez

$(11 + 48r + 36r^2 - 144r^3 - 30r^4 + 336r^5 - 348r^6 + 144r^7 - 21r^8)/32$. Például ez $7739/8192 \approx 0.94$, ha $r = 1/2$.

73. Tekintsük a következő Horn-klózokat: $1\wedge 2 \Rightarrow 3$, $1\wedge 3 \Rightarrow 4$, $\ldots$, $1\wedge(n-1) \Rightarrow n$, $1\wedge n \Rightarrow 2$ és $i\wedge j \Rightarrow 1$ minden $1 < i < j \le n$ esetén. Tegyük fel, hogy $|Z| > 1$ egy dekompozícióban és legyen i a legkisebb olyan szám, amelyre $x_i \in Z$. Továbbá legyen j a legkisebb olyan egész, amelyre $j > i$ és $x_j \in Z$. Ekkor nem lehet $i > 1$, ugyanis ekkor $i\wedge j \Rightarrow 1$ teljesülne. Ebből már következik, hogy $i = 1$ és $x_j \in Z$ teljesül minden $2 \le j \le n$ esetén.

74. Tegyük fel, hogy már tudjuk, hogy nem létezik olyan nemtriviális dekompozíció, amire $x_1 \in Z$, $\cdots$ vagy $x_{i-1} \in Z$ teljesülne, és kezdetben $i = 1$. Azt reméljük, hogy j és m ügyes megválasztásával $x_i \in Z$-t is kizárhatjuk. Az $i\wedge j \Rightarrow m$ alakú Horn-klózok $j \Rightarrow m$ alakú Krom-klózokká egyszerűsödnek, ha i igaz. Tarjan mélységi kereső algoritmusával akarjuk megkeresni az erős komponenseket egy olyan irányított gráfban, amelyben a $j \Rightarrow m$ élek talán léteznek, talán nem.

Amikor a j csúcsból kiinduló éleket vizsgáljuk, először próbáljuk $m = 1, \ldots, m = i-1$-et. Ha bármelyik $i\wedge j \Rightarrow m$ implikáció fennáll, akkor j-t és az összes ősét kitörölhetjük az i-hez tartozó irányított gráfból. Különben vizsgáljuk meg, hogy teljesül-e $j \Rightarrow m$ valamelyik már kitörölt m csúcsra. Ha ez sem sikerül, akkor tekintsük a még nem vizsgált m csúcsokat. Végezetül, ha az előzőek még nem voltak elegendőek, próbálkozzunk olyan m csúcsokkal, amelyeket már megvizsgáltunk, kezdve azokkal, amik közel vannak a mélységi fa gyökeréhez.

Példaként $f(x) = (\det X) \mod 2$-re rendre a következőket kapjuk: $1\wedge 2 \not\Rightarrow 3$, $1\wedge 2 \Rightarrow 4$, $1\wedge 4 \Rightarrow 3$, $1\wedge 3 \Rightarrow 5$, $1\wedge 5 \Rightarrow 6$, $1\wedge 6 \Rightarrow 7$, $1\wedge 7 \Rightarrow 8$, $1\wedge 8 \Rightarrow 9$, $1\wedge 9 \Rightarrow 2$ (immár $i \leftarrow 2$); $2\wedge 3 \not\Rightarrow 1$, $2\wedge 3 \Rightarrow 4$, $2\wedge 4 \not\Rightarrow 1$, $2\wedge 4 \not\Rightarrow 5$, $2\wedge 4 \Rightarrow 6$, $2\wedge 6 \Rightarrow 1$ (ekkor 3, 4 és 6 is törlésre kerül a 2-höz tartozó gráfból), $2\wedge 5 \Rightarrow 1$ (és 5-öt is törölhetjük), $2\wedge 7 \not\Rightarrow 1$, $2\wedge 7 \Rightarrow 3$ (7 is törlésre kerül), $2\wedge 8 \Rightarrow 1$, $2\wedge 9 \Rightarrow 1$ (most már $i \leftarrow 3$); $3\wedge 4 \not\Rightarrow 1$, $3\wedge 4 \Rightarrow 2$, $3\wedge 5 \Rightarrow 1$, és így tovább.

75. Ez a függvény két helyen vesz fel 1 értéket és ez a két hely egymás komplemense. Így tehát nem dekomponálható, mégis az (58) párok *sohasem* lesznek rosszak, ha $n > 3$. Ezért bármely (Y, Z) felosztás alkalmas dekompozíciónak.

Hasonlóképpen, ha f felbontható (Y, Z)-nek megfelelően, az $f(x) \oplus S_{0,n}(x)$ nem dekomponálható függvény lényegében véve ugyanúgy fog viselkedni a tesztekben mint f. (Egy általános célú dekomponálhatóság tesztelőnek tartalmaznia kellene módszert a *közelítőleg dekomponálható függvények* kezelésére is.)

76. (a) Legyen $a_l = [i \ge l]$, ha $0 \le l \le 2^m$. Ahogy a 38(b) gyakorlatban már láttuk, a költség $\le 2t(m)$, és valójában a költség tovább csökkenthető $2^{m+1} - 2m - 2$-re $\Theta(m)$ mélység mellett. Mi több, az $[i \le j] = (\bar{\imath}_1 \wedge j_1) \vee ((i_1 \equiv j_1) \wedge [i_2 \ldots i_m \le j_2 \ldots j_m])$ függvény $4m - 3$ kapu segítségével kiértékelhető. $x \oplus y$ kiszámítása után, valamennyi z_l költsége $2^{m+1} + 1 = O(n/\log n)$.

(b) Itt a költség legfeljebb $C(g_0) + \cdots + C(g_{2^m}) \le (2^m + 1)(2^{2^m}/(2^m + O(m)))$ az L tétel szerint, ugyanis az összes g_l függvénynek 2^m bemenete van.

(c) Ha $i \le j$, akkor $z_l = x$ minden $l \le i$ esetén és $z_l = y$, ha $l > i$. Így kapjuk, hogy $f_i(x) = c_0 \oplus \cdots \oplus c_i$ és $f_j(y) = c_{j+1} \oplus \cdots \oplus c_{2^m}$. Ha $i > j$, akkor $z_l = y$ minden $l \le i$ esetén és $z_l = x$, ha $l > i$. Ebben az esetben pedig $f_j(y) = c_0 \oplus \cdots \oplus c_j$ és $f_i(x) = c_{i+1} \oplus \cdots \oplus c_{2^m}$ adódik.

(d) A $b_l = [j < l]$ függvények $0 \le l \le 2^m$ esetén $O(2^m)$ lépésben kiszámíthatóak, ahogy azt láttuk (a)-ban. Tehát F kiszámítható $(c_0, \ldots, c_{2^m})$-ből $O(2^m)$ további kapu segítségével. Tehát a költséget alapvetően a (b) lépés határozza meg nagy m értékek esetében.

(e) $a_0 = 1$, $a_1 = i$, $a_2 = 0$; $b_0 = 0$, $b_1 = j$, $b_2 = 1$; $d = [i \le j] = \bar{\imath} \vee j$; $m_l = a_l \oplus d$, $z_{l0} = x_0 \oplus (m_l \wedge (x_0 \oplus y_0))$, $z_{l1} = x_1 \oplus (m_l \wedge (x_1 \oplus y_1))$, ha $l = 0, 1, 2$; $c_0 = z_{01}$; $c_1 = z_{10} \wedge \bar{z}_{11}$; $c_2 = z_{20} \vee z_{21}$; $c'_l = c_l \wedge (d \equiv a_l)$, $c''_l = c_l \wedge (d \equiv b_l)$, ha $l = 0, 1, 2$; és végül $F = (c'_0 \oplus c'_1 \oplus c'_2) \vee (c''_0 \oplus c''_1 \oplus c''_2)$.

A teljes költség 29 nyilvánvaló egyszerűsítések után, ami természetesen elfogadhatatlan ilyen kis méretű példára. De az emberben felmerülhet a kérdés, hogy egy korszerű gépi optimalizáló képes lenne-e leegyszerűsíteni ezt a láncot csupán 5 kapura. Ez az eredmény speciális esete azoknak a sokkal általánosabb tételeknek, amik megtalálhatóak Uhlig cikkeiben [60, 61].

77. Tegyük fel, hogy a lánc legrövidebb lánc f_n, illetve $\bar{f}_n$ kiszámítására. Ekkor legyen $U_l = \{i \mid l = j(i)$ vagy $l = k(i)\}$ azon lépések indexeinek halmaza, amelyek „felhasználják” az x_l változót és legyen $u_l = |U_l|$. Legyen továbbá $t_i = 1$, ha $x_i = x_{j(i)} \vee x_{k(i)}$, különben pedig $t_i = 0$. Meg fogjuk mutatni, hogy létezik olyan legfeljebb $r - 4$ hosszú lánc, ami kiszámítja az f_{n-1}, illetve $\bar{f}_{n-1}$ függvényt a következő ötlet felhasználásával: Ha valamely m esetén az x_m változó értéke 0-ra, illetve 1-re változik, az U_m-hez tartozó lépések törlésével és a többi lépés megfelelő módosításával kaphatunk egy láncot f_{n-1}, illetve $\bar{f}_{n-1}$ kiszámítására. Továbbá, ha $x_i = x_{j(i)} \circ x_{k(i)}$ és ha $x_{j(i)}$-ről vagy $x_{k(i)}$-ről tudjuk, hogy megegyezik t_i-vel, ha x_m 0-ra vagy 1-re változik, akkor törölhetjük az U_i-hez tartozó lépéseket is. (A tárgyalás során m végig egy tetszőleges $1 \leq m \leq n$ indexet jelöl.)

1. eset: $u_m = 1$ valamely m-re. Ez az eset nem fordulhat elő legrövidebb lánc esetén. Ugyanis, ha x_m egyetlen felhasználása $x_i = \bar{x}_m$, a lépés törlésével $f_n \leftrightarrow \bar{f}_n$-t kapnánk, és különben $x_1, \ldots, x_{m-1}, x_{m+1}, \ldots, x_n$ értékét megváltoztathatnánk oly módon, hogy x_i független legyen x_m-től, ellentmondva ezzel $x_{n+r} = f_n$ vagy $\bar{f}_n$-nak. Tehát minden változót legalább kétszer felhasználnunk.

2. eset: $x_l = \bar{x}_m$ valamely l-re és m-re, ahol $u_m > 1$. Ekkor $x_i = x_l \circ x_k$ valamely i-re és k-ra, és x_m értékét $\bar{t}_i$-ra módosíthatjuk, így x_i-t téve függetlenné x_k-tól. Ekkor az U_m, U_l és U_i-hez tartozó lépések törlésével legalább 4 lépést eltávolíthatunk, kivéve ha $u_l = u_i = 1$, $u_m = 2$ és $x_j = x_m \circ x_i$, de ebben az esetben U_j is törölhető.

3. eset: $u_m \geq 3$ valamely m-re és nem felel meg a 2. esetnek. Ha $i, j, k \in U_m$ és $i < j < k$, módosítsuk x_m értékét t_k-ra és töröljük az i, j, k, U_k lépéseket.

4. eset: $u_1 = u_2 = \cdots = u_n = 2$, és nem tartozik a 2. esethez. Feltehetjük, hogy az első lépés $x_1 = x_1 \circ x_2$ és hogy $x_l = x_1 \circ x_k$ valamely $k < l$ esetén.

4.1. eset: $k > 0$. Ekkor $k > 1$. Ha $u_k = 1$, legyen $x_1 \leftarrow t_l$ és töröljük az 1, k, l, U_l lépéseket. Különben legyen $x_2 \leftarrow t_1$, így $x_k = \bar{t}_l$-nak kell teljesülnie és törölhetjük az 1, k, l, U_k lépéseket.

4.2. eset: $x_l = x_1 \circ x_m$. Ekkor $m = 2$-nek kell teljesülnie, ha ugyanis $m > 2$ teljesülne, elvégezve az $x_2 \leftarrow t_1$, $x_m \leftarrow t_l$ módosításokat az x_r változót függetlenné tehetnénk x_1-től. Továbbá feltehetjük, hogy $x_1 = x_1 \wedge x_2$ és $x_2 = x_1 \vee x_2$. Elvégezve $x_1 \leftarrow$ 0-át törölhetjük az U_0 és U_1 lépéseket, elvégezve $x_1 \leftarrow$ 1-et pedig törölhetjük az U_0 és U_2 lépéseket. Így készen is vagyunk, hacsak nem teljesül $u_1 = u_2 = 1$.

Ha $x_p = \bar{x}_1$, legyen $x_1 \leftarrow 0$ és töröljük az 1, 2, p, U_p lépéseket; ha $x_q = \bar{x}_2$, legyen $x_1 \leftarrow 1$ és töröljük az 1, 2, q, U_q lépéseket. Különben $x_p = x_1 \circ x_u$ és $x_q = x_2 \circ x_v$, ahol x_u és x_v nem függ x_1-től vagy x_2-től. De ez lehetetlen, ugyanis ez lehetővé tenné, hogy $x_3, \ldots, x_n$ értékét úgy módosítsuk, hogy $x_u = t_p$ teljesüljön, ez követően $x_2 \leftarrow 1$ segítségével x_r-t függetlenné tehetnénk x_1-től.

Lásd [43, 44] Hasonló bizonyítással Redkin megmutatta, hogy az ‘$x_1 \ldots x_n < y_1 \ldots y_n$’ és ‘$x_1 \ldots x_n = y_1 \ldots y_n$’ függvényekhez tartozó legrövidebb AND-OR-NOT láncok hossza rendre $5n-3$ és $5n-1$.

78. Lásd Paul cikkét [39, 427–430. oldal]. Nevezzük az y_k változót *aktívnak*, ha $k \in S$ teljesül. Feltehetjük, hogy a lánc normalizált és hogy $\|S\| > 1$. A bizonyítás hasonló Redkin bizonyításához a 77. gyakorlat megoldásában: *1. eset:* Valamely aktív y_k-t többször is felhasználunk. Ekkor $y_k \leftarrow 0$ legalább két lépést megtakarít és egy olyan láncot eredményez, ami egy $\|S\| - 1$ aktív értékkel rendelkező függvényt számít ki.

2. eset: Az egyik aktív y_k csak egyetlen AND-kapuban jut szerephez. Ekkor $y_k \leftarrow 0$ segítségével legalább két lépés törölhető, ha az AND nem az utolsó lépés. De ez nem is lehet az utolsó lépés, ugyanis $y_k = 0$ esetén a végeredmény független lenne az összes többi aktív y_j-től.

3. eset: Ugyanúgy bizonyítható, ahogyan a 2. esetben, csak VAGY, NEM-DE vagy DE-NEM kapuval. Az $y_k \leftarrow c$ értékadás segítségével elérhető a kívánt hatás egy megfelelő c konstanssal.

4. eset: Mint a 2. esetben, csak XOR kapuval. A kapu nem lehet az utolsó, ugyanis a végeredménynek függetlennek kell lennie y_k-tól, ha $(x_1 \ldots x_m)_2$ egy másik aktív y_j-re hivatkozik. Ekkor törölhetünk két lépést, ha y_k értékét a XOR *másik* bemenete által meghatározott függvényre módosítjuk.

79. (a) Tegyük fel, hogy a költség $r < 2n - 2$, ahol $n > 1$. Ha minden változót pontosan egyszer használunk fel, bármely két levélnek barátnak kell lennie. Így tehát néhány változót legalább kétszer kellett felhasználnunk. Egy ilyen változót eltávolítva a kapott $n - 1$ változós lánc költsége $\leq r - 2$ és nem tartalmaz barátokat. (Egyébként a költség legalább $2n - 1$, ha minden változót legalább kétszer használunk fel, ugyanis a változóknak legalább $2n$ használatát kell összekapcsolni a láncban.)

(b) Vegyük észre, hogy $S_{0,n} = \bigwedge_{u@—@v}(u \equiv v)$ ha az $u—v$ élek szabad fát határoznak meg $\{x_1, \ldots, x_n\}$ felett. Ebből következik, hogy sokféleképpen el lehet érni a $2n - 3$ költséget.

Minden $r < 2n - 3$ költségű láncra $n > 2$-nek kell teljesülnie és az ilyen láncoknak tartalmazniuk kell u, v barátokat. Átnevezéssel és a közvetlen eredmények komplementálásával feltehetjük, hogy $u = 1$, $v = 2$ és hogy $f(x_1, \ldots, x_n) = g(x_1 \circ h(x_3, \ldots, x_n), x_2, \ldots, x_n)$, ahol $\circ$ vagy az $\wedge$ vagy a $\oplus$ művelet.

1. eset: $\circ$ az AND művelet. Ekkor $h(0, \ldots, 0) = h(1, \ldots, 1) = 1$-nek kell teljesülnie, különben $f(x_1, x_2, y, \ldots, y)$ nem függne x_1-től. Ezért $f(x_1, \ldots, x_n) = h(x_3, \ldots, x_n) \wedge g(x_1, x_2, \ldots, x_n)$ kiszámítható egy olyan változatlan költségű lánccal, amiben 1 és 2 barátok és az út közöttük rövidebb mint eddig volt.

2. eset: $\circ$ a XOR művelet. Ekkor $f = f_0 \vee f_1$, ahol $f_0(x_1, \ldots, x_n) = (x_1 \equiv h(x_3, \ldots, x_n)) \wedge g(0, x_2, \ldots, x_n)$ és $f_1(x_1, \ldots, x_n) = (x_1 \oplus h(x_3, \ldots, x_n)) \wedge g(1, x_2, \ldots, x_n)$. De $f = S_{0,n}$-nek csak két prím implikánsa van, így csak négy lehetőség van:

2a. eset: $f_0 = f$. Ekkor felcserélhetjük $x_1 \oplus h$-t 0-val, hogy egy $\leq r-2$ költségű láncot kapjunk a $g(0, x_2, \ldots, x_n) = S_{0,n-1}(x_2, \ldots, x_n)$ függvényre.

2b. eset: $f_1 = f$. Hasonlóképpen mint az előző esetben.

2c. eset: $f_0(x) = x_1 \wedge \cdots \wedge x_n$ és $f_1(x) = \bar{x}_1 \wedge \cdots \wedge \bar{x}_n$. Ebben az esetben a következőknek kell teljesülniük: $g(0, x_2, \ldots, x_n) = x_2 \wedge \cdots \wedge x_n$ és $g(1, x_2, \ldots, x_n) = \bar{x}_2 \wedge \cdots \wedge \bar{x}_n$. Helyettesítve h-t 1-gyel egy olyan lánchoz jutunk, ami az f függvényt kevesebb mint r lépésben számítja ki.

2d. eset: $f_0(x) = \bar{x}_1 \wedge \cdots \wedge \bar{x}_n$ és $f_1(x) = x_1 \wedge \cdots \wedge x_n$. Hasonlóképpen járhatunk el, ahogy az előző esetben.

A fenti egyszerűsítések ismételt alkalmazása ellentmondásra vezet. Ugyanígy megmutatható, hogy $C(S_0 S_n) = 2n - 2$. Lásd még [48].

80. [57] Az általánosság megsértése nélkül feltehetjük, hogy $a_0 = 0$ és hogy a lánc normalizált. Definiáljuk U_l-t és u_l-t, ahogyan a 77. gyakorlat megoldásában is tettük. A szimmetria miatt feltehetjük, hogy $u_1 = \max(u_1, \ldots, u_n)$.

Ekkor $u_1 \geq 2$-nek kell teljesülnie. Ugyanis, ha $u_1 = 1$ teljesülne, akkor feltehetnénk, hogy $x_{n+1} = x_1 \circ x_2$ és így kettő a következő három függvény közül egyenlő lenne: $S_\alpha(0, 0, x_3, \ldots, x_n) = S_{\alpha''}$, $S_\alpha(0, 1, x_3, \ldots, x_n) = S_{'\alpha'}$, $S_\alpha(1, 1, x_3, \ldots, x_n) = S_{''\alpha}$. De ekkor S_α paritás függvény lenne vagy $S_{'\alpha'}$ lenne konstans. Így x_1-nek 0-át adva értékül törölhetjük az U_1-hez tartozó kapukat, ami egy legalább 2-vel rövidebb láncot eredményez $S_{\alpha'}$-re. Ebből következik, hogy $C(S_\alpha) \geq C(S_{\alpha'}) + 2$. Hasonlóan x_1-nek 1-et adva értékül bizonyíthatjuk, hogy $C(S_\alpha) \geq C(S_{'\alpha}) + 2$.

A további vizsgálatokhoz három esetet kell tekinteni:

1. eset: $u_1 \geq 3$. Legyen $x_1 = 0$, ez bizonyítja, hogy $C(S_\alpha) \geq C(S_{\alpha'}) + 3$.

2. eset: $U_1 = \{i, j\}$ és a $\circ_j$ irányító művelet (azaz ÉS, DE-NEM, NEM-DE vagy VAGY). x_1 értékének egy megfelelő konstansra módosítása lehetővé teszi, hogy töröljük az $U_1 \cup U_j$ lépéseket. Vegyük észre, hogy optimális lánc esetén $i \notin U_j$. Így tehát vagy $C(S_\alpha) \geq C(S_{\alpha'}) + 3$ vagy $C(S_\alpha) \geq C(S_{'\alpha}) + 3$ teljesül.

3. eset: $U_1 = \{i, j\}$ és $\circ_i = \circ_j = \oplus$. Feltehetjük, hogy $x_i = x_1 \oplus x_2$ és $x_j = x_1 \oplus x_k$. Ha $u_j = 1$ és $x_l = x_j \oplus x_p$, akkor átszervezhetjük a láncot $x_j = x_k \oplus x_p$, $x_l = x_1 \oplus x_j$ felhasználásával, így feltehetjük, hogy $u_j \neq 1$ vagy $x_l = x_j \circ x_p$ teljesül $\circ$ irányító művelettel. Ha $U_2 = \{i, j'\}$, akkor ugyanígy feltehetjük, hogy $x_{j'} = x_2 \oplus x_{k'}$ és hogy $u_{j'} = 1$ vagy $x_{l'} = x_{j'} \circ' x_{p'}$ teljesül egy $\circ'$ irányító művelettel. Továbbá a szimmetria miatt feltehetjük, hogy x_j nem függ $x_{j'}$-től.

Ha x_k nem függ x_i-től, akkor legyen $f(x_3, \ldots, x_n) = x_k$, különben pedig legyen $f(x_3, \ldots, x_n)$ értéke x_k, ha $x_i = 1$. Elvégezve az $x_1 = f(x_3, \ldots, x_n)$ és $x_2 = \bar{f}(x_3, \ldots, x_n)$ értékadásokat, vagy ezek fordítottját, x_i és x_j konstanssá tehető, és így láncot kaphatunk a nemkonstans $S'_{\alpha'}$ függvényre. Tulajdonképpen $u_j = 1$ esetén biztosíthatjuk, hogy x_l konstans legyen. Azt állítjuk, hogy ennek a láncnak legalább öt kapuja (beleértve x_i-t és x_j-t is) törölhető. Így $C(S_\alpha) \geq C(S'_{\alpha'})+5$. Az állítás nyilván igaz, ha $|U_i \cup U_j| \geq 3$.

$|U_i \cup U_j| > 1$-nek fenn kell állnia. Különben $p = i$ teljesülne és x_k nem függne x_i-től, így S_α független lenne x_1-től a mi x_2-nk mellett. Így tehát $|U_i \cup U_j| = 2$.

3a eset: $U_j = \{l\}$. Ekkor x_l konstans és törölhetjük az x_i, x_j és $U_i \cup U_j \cup U_l$ lépéseket. Ha az utóbbi halmaz csak kételemű, akkor $x_q = x_i \circ x_l$ is konstans és U_q is törölhető. Mivel $S'_{\alpha'}$ nem konstans, a kimenet kapuját nem fogjuk törölni.

3b eset: $U_i \subseteq U_j$, $|U_j| = 2$. Ekkor $x_q = x_i \circ x_j$ valamilyen q-ra és az x_i, x_j és $U_j \cup U_q$ lépések törölhetőek. Az állítást bebizonyítottuk.

(b) Indukcióval, $C(S_k) \geq 2n + \min(k, n-k) - 3 - [n = 2k]$, minden $0 < k < n$-re; $C(S_{\geq k}) \geq 2n+\min(k, n+1-k)-4$, ha $1 < k < n$. Az egyszerű esetek $C(S_0) = C(S_n) = C(S_{\geq 1}) = C(S_{\geq n}) = n-1$; $C(S_{\geq 0}) = 0$. (A 7.9. és 7.10. ábrák szerint ezek a korlátok optimálisak $k = \lceil n/2 \rceil$-re, ha $n \leq 5$. Egyetlen ismert eredmény sem mond ellent a $C(S_k) = C(S_{\geq k})$ ha $k \geq n/2$ feltevésnek.)

81. Ha egy változót többször is használunk, akkor legyen az értéke konstans, ezáltal csökkentve n értékét eggyel és c értékét legalább 2-vel. Ellenkező esetben az első műveletnek x_1-re is kell vonatkoznia, ugyanis $y_1 = x_1$ az egyetlen kimenet, ami nem igényel számolást. Ekkor x_1 konstansra módosítása csökkenti n értékét eggyel, c-ét és d-ét pedig legalább eggyel-eggyel. Lásd még [53].

82. (62) hamis.

(63) a következőképpen hangzik: "Minden m számra létezik egy olyan n szám, hogy $m < n+1$". Ez igaz, ugyanis az $m = n$ választás kielégíti az állítást.

64 hamis, ha $n = 0$ vagy $n = 1$, mert ezekben a formulákban a számoknak nemnegatív egészeknek kell lenniük.

(65) azt mondja ki, hogy ha b értéke meghaladja a értékét legalább kettővel, akkor megadható egy ab szám közöttük. Ez természetesen igaz, az $ab = a + 1$ szám megfelel a feltételnek.

(66)-ot megindokoltuk már a szövegben és ez is igaz. Vegyük észre, hogy '$\wedge$' magasabb precedenciájú mint '$\vee$', '$\equiv$' mint '$\Leftrightarrow$', ahogyan '$+$' mint '$\geq$' és '$<$' mint '$\wedge$' (65)-ben. Ezen konvenciók használatával csökkenthetjük a zárójelek számát L mondataiban. (67) azt állítja, hogy ha A legalább egy n elemet tartalmaz, akkor tartalmaznia kell egy m minimális elemet is, azaz egy olyan elemet, ami kisebb vagy egyenlő minden más eleménél. Igaz.

(68) hasonló állítást fogalmaz meg, csak m most maximális elem. Ismét igaz, ugyanis minden halmaz véges.

(69) egy olyan P halmazt kér, amelyre teljesül $[0 \in P] = [3 \notin P]$, $[1 \in P] = [4 \notin P]$, $\ldots$, $[999 \in P] = [1002 \notin P]$, $[1000 \in P] \neq [1003 \notin P]$, $[1001 \in P] \neq [1004 \notin P]$ stb. Ez igaz akkor (és csak akkor), ha $P = \{x \mid x \mod 6 \in \{1, 2, 3\}$ és $0 \leq x < 1000\}$.

Végül, a $\forall n\, (n \in C \Leftrightarrow n + 1 \in C)$ részformula 70-ben egy másik megfogalmazása annak, hogy $C = \emptyset$, ugyanis C véges. Következésképpen a $\forall A \forall B$-t követő zárójelezett formula egy trükkös megfogalmazása annak, hogy $A = \emptyset$ és $B \neq \emptyset$. (Stockmeyer és Meyer ezt a trükköt arra használták, hogy lerövidítsék L hosszú részformulákat többszörösen is tartalmazó állításait.) A (70) állítás tehát igaz, mert az üres halmaz nem lehet egyenlő egy nemüres halmazzal.

83. Feltehetjük, hogy a lánc normalizált. Legyenek az irányító lépések y_1, $\ldots$, y_p. Ekkor $y_k = \alpha_k \circ \beta_k$ és $f = \alpha_{p+1}$, ahol α_k és β_k az $\{x_1, \ldots, x_n, y_1, \ldots, y_{k-1}\}$ valamely részhalmazainak $\oplus$-jai és legfeljebb $n + k - 2$ $\oplus$ szükséges a kiszámításukhoz, ha először az egyszerű termeket kombináljuk.

Így $C(f) \le p + \sum_{k=1}^{p+1}(n+k-2) = (p+1)(n+p/2) - 1$.

84. Az előző megoldásban használt érvelést alkalmazhatjuk ezúttal is, csak $\vee$ és $\wedge$ szerepel $\oplus$ helyett. Lásd még Alon és Boppana cikkét [2, 15–16].

85. (a) Egy egyszerű számítógépes programmal kiszámítható, hogy 13744 legitim közülük és 19024 nem. (Egy ilyen illegitim családnak legalább 8 tagja kell, hogy legyen. Ilyen például $\{\texttt{00}, \texttt{0f}, \texttt{33}, \texttt{55}, \texttt{ff}, \texttt{15}, \texttt{3f}, \texttt{77}\}$. Valóban, ha az $x_1 \vee x_2$ (`3f`), $x_2 \vee x_3$ (`77`) és $(x_1 \vee x_2) \wedge x_3$ (`15`) függvényeket tartalmazza egy L legitim család, akkor az $x_2 \sqcup \texttt{15} = \texttt{33@|@15} = \texttt{37}$ függvényt szintén tartalmaznia kell.)

(b) A projekciós és konstans függvényeket nyilván tartalmazza. Legyen $A^* = \bigcap\{B \mid B \supseteq A$ és $B \in \mathcal{A}\}$, vagy $A^* = \infty$, ha nem létezik ilyen B halmaz. Ekkor $\lceil A\rceil \sqcap \lceil B\rceil = \lceil A \cap B\rceil$ és $\lceil A\rceil \sqcup \lceil B\rceil = \lceil (A \cup B)^*\rceil$.

(c) Rövidítsük le a formulákat a következőféleképpen: $\hat{x}_l \subseteq x_l \vee \bigvee_{i=n+1}^{l} \delta_i$, $x_l \subseteq \hat{x}_l \vee \bigvee_{i=n+1}^{l} \epsilon_i$, és bizonyítsunk indukcióval: Ha az l. lépés AND lépés, akkor $\hat{x}_l = \hat{x}_j \sqcap \hat{x}_k \subseteq \hat{x}_j \wedge \hat{x}_k \subseteq \big(x_j \vee \bigvee_{i=n+1}^{l} \delta_i\big) \wedge \big(x_k \vee \bigvee_{i=n+1}^{l} \delta_i\big) = x_l \vee \bigvee_{i=n+1}^{l} \delta_i$; $x_l = x_j \wedge x_k \subseteq \big(\hat{x}_j \vee \bigvee_{i=n+1}^{l-1} \epsilon_i\big) \wedge \big(\hat{x}_k \vee \bigvee_{i=n+1}^{l-1} \epsilon_i\big) = (\hat{x}_j \wedge \hat{x}_k) \vee \bigvee_{i=n+1}^{l-1} \epsilon_i$, és $\hat{x}_j \wedge \hat{x}_k = \hat{x}_l \vee \epsilon_l$. Ugyanígy érvelhetünk, ha az l. lépés OR lépés.

86. (a) Ha S egy r-család, amit tartalmaz az S' $(r+1)$-család, akkor nyilván $\Delta(S) \subseteq \Delta(S')$.

(b) A skatulya elv alapján, ha S egy r-család, akkor $\Delta(S)$-nek az összes rész u és v elemét tartalmaznia kell. És, ha $\Delta(S) = \{u, v\}$, akkor u — v biztosan fennáll.

(c) Az eredmény nyilvánvaló, ha $r = 1$. Az „erős" tulajdonság miatt legfeljebb $r-1$ olyan él lehet, ami illeszkedik egy adott u csúcsra. És, ha u — v, az $\{u, v\}$-ből *kiinduló* élek erősen $(r-1)$-zártak, így indukcióval megmutatható, hogy legfeljebb $(r-2)^2$ lehet belőlük. Tehát összesen legfeljebb $1 + 2(r-2) + (r-2)^2$ él lehet.

(d) Igen, a 85(b) gyakorlat miatt, ha $r > 1$. Ugyanis az erősen r-zárt gráfok zártak a metszetre. Minden legfeljebb 1 élű gráf erősen r-zárt, ha $r > 1$, hiszen nem találhatunk bennük olyan r-családokat, melyek élei diszjunktak.

(e) Összesen $\binom{n}{3}$ $x_{ij} \wedge x_{ik} \wedge x_{jk}$ háromszög létezik, melyek közül csak $n-2$ fordul elő $\hat{f}$ valamely x_{uv} termjében. Így legfeljebb $(r-1)^2(n-2)$ háromszög mintermje szerepel $\hat{f}$-ban, és a többit pedig valamely $\epsilon_i = \hat{x}_i \oplus (\hat{x}_{j(i)} \wedge x_{k(i)})$ függvény tartalmazza. Egy ilyen term $T = (\lceil G\rceil \sqcap \lceil H\rceil) \oplus (\lceil G\rceil \wedge \lceil H\rceil) = (\lceil G\rceil \wedge \lceil H\rceil) \wedge \overline{\lceil G \cap H\rceil}$ alakban felírható, ahol G és H erősen r-zártak. Be fogjuk bizonyítani, hogy T legfeljebb $2(r-1)^3$ háromszöget tartalmaz.

Egy $x_{ij} \wedge x_{ik} \wedge x_{jk}$ háromszögnek T-ben fel kell használnia valamilyen változót (mondjuk x_{ij}-t) $\lceil G\rceil$-ből és valamilyen változót (mondjuk x_{ik}-t) $\lceil H\rceil$-ból, de egyetlen változót sem használhat fel $\lceil G \cap H\rceil$-ból. Legfeljebb $(r-1)^2$-féleképpen lehet ij-t megválasztani és legfeljebb $2(r-1)$-féleképpen k-t, ugyanis H-nak legfeljebb $r-1$ éle érinti i-t és legfeljebb $r-1$ éle j-t.

(f) Színezzük az 1 csúcsot pirosra, a többi csúcsot pedig pirosra vagy kékre. Egy u és v csúcsot pedig kössön össze egy él pontosan akkor, ha eltérő színűek. Ily módon 2^{n-1} teljes páros gráfot kaphatunk. A 85(c) gyakorlat első formulája alapján minden ilyen gráfra a B mintermet egy $T = \delta_i = \hat{x}_i \oplus (\hat{x}_{j(i)} \vee x_{k(i)}) = \lceil (G \cup H)^*\rceil \wedge \overline{\lceil G \cup H\rceil}$ termnek tartalmaznia kell. (Például, ha $n = 4$ és a $(2, 3, 4)$ csúcsok színe (piros, kék, kék), akkor $B = \bar{x}_{12} \wedge x_{13} \wedge x_{14} \wedge x_{23} \wedge x_{24} \wedge \bar{x}_{34}$.) Egy B mintermet pontosan akkor tartalmaz T, ha B színezésével $(G \cup H)^*$ valamely éle eltérő színű csúcsokra illeszkedik, míg $G \cup H$ minden éle egyszínű. Bebizonyítjuk, hogy T legfeljebb $2^{n-r}r^2$ ilyen B termet tartalmazhat.

Legyen G egy tetszőleges gráf és $T = \lceil G^*\rceil \wedge \overline{\lceil G\rceil}$. A következő nem túl hatékony algoritmussal megtalálható G^*: Ha létezik egy S r-family, amire $|\Delta(S)| < 2$, akkor megállunk és legyen $G^* = \infty$. Különben, ha $\Delta(S) = \{u, v\}$ és $u \not\!\!-\, v$, vegyük hozzá G-hez az u — v élt és így tovább.

Legfeljebb 2^{n-r} kétrétű B mintermre lesznek az $\{u_j, v_j\}$-k $(1 \le j \le r)$ egyszínűek, ha $|\Delta(S)| < 2$. És. ha $\Delta(S) = \{u, v\}$, akkor 2^{n-r-1}-re lesznek az $\{u_j, v_j\}$-k egyszínűek és az $\{u, v\}$ kétszínű. Így tehát azt akarjuk megmutatni, hogy G-re az algoritmus kevesebb mint $2r^2$ iterációt végez, ha G erősen r-zárt.

$k \ge 1$-re legyen u_k — v_k az első új él, amit G-hez hozzáveszünk és $\{u_j, v_j\}$-ből $(1 \le j < k)$

indul ki. Legfeljebb r ilyen él létezik az erős tulajdonság miatt és mindegyiket legfeljebb $2r-3$ u_j-t vagy v_j-t érintő új él követi. Így összesen legfeljebb $r(2r-2)+1<2r^2$ lépésben megtaláljuk G^*-ot.

(g) A 84-es gyakorlat szerint $q<\binom{p}{2}+(p+1)\binom{n}{2}$. Így vagy $2(r-1)^3p\geq\binom{n}{3}-(r-1)^2(n-2)$ vagy pedig $\binom{p}{2}+(p+1)\binom{n}{2}>2^{r-1}/r^2$ teljesül. Mindkét p-re vonatkozó alsó korlát

$$\frac{1}{12}\Big(\frac{n}{6\lg n}\Big)^3\Big(1+O\Big(\frac{\log\log n}{\log n}\Big)\Big)\qquad\text{ha}\qquad r=\Big\lceil\lg\Big(\frac{n^6}{186624(\lg n)^4}\Big)\Big\rceil.$$

Noga Alon és Ravi B. Boppana [2] így bizonyították be többek között az $\wedge$-k számára az $\Omega(n/\lg n)^s$ alsó korlátot olyan monoton láncok esetében, amik egy adott G-ről eldöntik, hogy tartalmaz-e $s\geq 3$ rögzített méretű klikket.

87. X^3 legfeljebb n^2 bejegyzést tartalmaz, ha X egy 0–1 mátrix. Megadható olyan Boole-lánc, ami $O(n^{\lg 7}(\lg n)^2)$ kapu segítségével megvalósítja Strassen 4.6.4–(36) mátrixszorzó algoritmusát modulo $2^{\lfloor\lg n^2\rfloor+1}$ egészeken.

88. Összesen 1,422,564 ilyen függvény létezik. Ezeket 716 osztályba sorolhatjuk be, figyelembe véve a változók permutációit is. Az L algoritmus és a fejezet többi módszere mind kiterjeszthető háromváltozós műveletekre is, és a következő eredményekhez juthatunk az optimális csak-medián számításokra vonatkozóan:

$C(f)$	Osztá-lyok	Függ-vények	$C_m(f)$	Osztá-lyok	Függ-vények	$L(f)$	Osztá-lyok	Függ-vények	$D(f)$	Osztá-lyok	Függ-vények
0	1	7	0	1	7	0	1	7	0	1	7
1	1	35	1	1	35	1	1	35	1	1	35
2	2	350	2	2	350	2	2	350	2	13	5670
3	9	3885	3	9	3885	3	8	3745	3	700	1416822
4	48	42483	4	48	42483	4	38	35203	4	1	30
5	201	406945	5	188	391384	5	139	270830	5	0	0
6	353	798686	6	253	622909	6	313	699377	6	0	0
7	99	169891	7	69	134337	7	176	367542	7	0	0
8	2	282	8	2	2520	8	34	43135	8	0	0
9	0	0	9	0	0	9	3	2310	9	0	0
10	0	0	10	0	0	10	0	0	10	0	0
11	0	0	∞	143	224654	11	1	30	11	0	0

Amarel,, Cooke és Winder arra következtettek [3], hogy a 9 műveletből álló

$$\langle x_1x_2x_3x_4x_5x_6x_7\rangle=\big\langle x_1\langle\langle x_2x_3x_5\rangle\langle x_2x_4x_6\rangle\langle x_3x_4x_7\rangle\rangle\langle\langle x_2x_5x_6\rangle\langle x_3x_5x_7\rangle\langle x_4x_6x_7\rangle\rangle\big\rangle$$

formula a legjobb módja a 7-medián 3-mediánokkal való kiszámításának. De a „bűvös”

$$\big\langle x_1\langle x_2\langle x_3x_4x_5\rangle\langle x_3x_6x_7\rangle\rangle\langle x_4\langle x_2x_6x_7\rangle\langle x_3x_5\langle x_5x_6x_7\rangle\rangle\rangle\big\rangle$$

formulával csak 8 műveletre van szükség. Mi több, a legrövidebb lánc csak hét lépésből áll:

$$\langle x_1x_2x_3x_4x_5x_6x_7\rangle=\big\langle x_1\langle x_2\langle x_5x_6x_7\rangle\langle x_3\langle x_5x_6x_7\rangle x_4\rangle\rangle\langle x_5\langle x_2x_3x_4\rangle\langle x_6\langle x_2x_3x_4\rangle x_7\rangle\rangle\big\rangle.$$

Az érdekes $f(x_1,\ldots,x_7)=(x_1\wedge x_2\wedge x_4)\vee(x_2\wedge x_3\wedge x_5)\vee(x_3\wedge x_4\wedge x_6)\vee(x_4\wedge x_5\wedge x_7)\vee(x_5\wedge x_6\wedge x_1)\vee(x_6\wedge x_7\wedge x_2)\vee(x_7\wedge x_1\wedge x_3)$ függvény, amely prím implikánsai megfelelnek egy 7 pontú projekciós síknak, az összes közül a legnehezebb: A minimális $L(f)=11$ hosszát és legkisebb $D(f)=4$ mélységét a következő figyelemre méltó formulával reprezentálva éri el:

$$\big\langle\langle x_1x_4\langle x_4x_5x_6\rangle\rangle\langle x_3x_6\langle x_1\langle x_2x_3x_7\rangle\langle x_2x_5x_6\rangle\rangle\rangle\langle x_2x_7\langle x_1\langle x_5x_2x_4\rangle\langle x_5x_3x_7\rangle\rangle\rangle\big\rangle.$$

És a következő még megdöbbentőbb lánc számítja ki optimálisan:

$$x_8 = \langle x_1 x_2 x_3 \rangle, \quad x_9 = \langle x_1 x_4 x_6 \rangle, \quad x_{10} = \langle x_1 x_5 x_8 \rangle, \quad x_{11} = \langle x_2 x_7 x_8 \rangle,$$
$$x_{12} = \langle x_3 x_9 x_{10} \rangle, \quad x_{13} = \langle x_4 x_5 x_{12} \rangle, \quad x_{14} = \langle x_6 x_{11} x_{12} \rangle, \quad x_{15} = \langle x_7 x_{13} x_{14} \rangle.$$

IRODALOMJEGYZÉK

[1] H. Aiken. Synthesis of electronic computing and control circuits. *Annals of the Computer Laboratory of Harvard Harvard University*, 27, 1951. MR1638794 (99c:05018). 9

[2] N. Alon, R. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987. MR0905147 (88j:68044). 57, 58

[3] S. Amarel, G. Cooke, R. Winder. *IEEE Transactions on Computers*, EC-13:4–13, 1964. 58

[4] R. L. Ashenhurst. *Annals of Computation Laboratory of Harvard University*, 29:74–116, 1957. 23, 25, 26

[5] H. P. Babbage. *Babbage's Calculating Engines*. The MIT Press, 20044 (az első kiadás 1889-ben jelentette meg Londonban az F. N. Spon Kiadó). 21, 46

[6] C. P. Babbage. *Passages from the Life of a Philosopher*. Rutgers University Press, 1994 (az első kiadás 1864-ben jelent meg). 21

[7] J. R. Büchi, C. C. Elgot. *Zeitschrift für mathematische Logik und Grudlagen der Mathematik*, 6:66–92, 1960. 29

[8] E. R. Berlekamp, J. H. Conway, R. K. Guy. *Winning Ways for your Mathematical Plays. Volume 3*. AK Peters, Natick, 2003 (második kiadás). MR2006327 (2004f:91043). 51

[9] N. Blum. A Boolean function requiring $3n$ network size. *Theoretical Computer Science*, 28(3):337–345, 1984. MR0742295 (86g:68053). 28

[10] E. Boros, V. A. Gurvich, P., T. Ibaraki. Decomposability of partially defined Boolean functions. Partitioning and decomposition in combinatorial optimization. *Discrete Applied Mathematics*, 62(1–3):51–75, 1995. MR1355573 (96j:68087). 26

[11] R. Brayton, G. Hachtel, A. Sangiovanni-Vincentelli. Multilevel logic synthesis. *Proceedings of the IEEE*, 78(2):264–300, 1990. 27

[12] R. Brent, H. T. Kung. A regular layout for parallel adders. *IEEE Transactions on Computers*, 31(3):260–264, 1982. MR0656129 (83i:05012). 46

[13] H. A. Curtis. A generalized tree circuit. *Journal of the ACM*, 8:484–496, 1961. MR0143678 (26 #1231). 26

[14] H. A. Curtis. Simplified decomposition of Boolean functions. *IEEE Transactions on Computers*, C-25:291–335, 1976. MR0469531 (57 #9316). 26

[15] H. M. Curtis. *A New Approach to the Design of Switching Circuits*. Van Nostrand Reinhold, New York, 1962. 2

[16] C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–51, 1961. MR0139530 (25 #2962). 29

[17] P. Fenwick. A fast-carry adder with CMOS transmission gates. *The Computer Journal*, 30(1):77–79, 1987. 46

[18] B. Finikov. On a family of classes of functions in the logic algebra and their realization in the class of π-schemes (oroszul). *Doklady Akademii Nauk SSSR (Novaja Serija)*, 115:247–248, 1957. 30, 39

[19] M. Gardner. *Hexaflexagons and Other Mathematical Diversions*. University of Chicago Press, Chicago, 1988. 51

[20] M. Gardner. *The Scientific American Book of Mathematical Puzzles & Diversions*. Simon & Schuster, New York, 1959. 52

[21] L. Hellerman. A catalog of three-variable or-invert and and-invert logical circuits. *IEEE Transactions on Computers*, EC-12(3):198–223, 1963. 10

[22] S. L. Hight. Complex disjunctive decomposition of incompletely specified Boolean functions. *IEEE Transactions on Computers*, C-22(1):103–110–344, 1973. 26

[23] R. Karp. Functional decomposition and switching circuit design. *Journal of Society for Industrial and Applied Mathematics*, 11:291–335, 1963. MR0155728 (27 #5662). 26

[24] W. Keister. Tic-tac toe gép, 1940. A gép a *Museum of Science and Industry in Chicago* múzeumban látható. 21

[25] V. M. Khrapchenko. *Problemy Kibernetiki*, 19:107–122, 1967. 46

[26] D. E. Knuth, R. L. Graham, O. Patashnik. *Concrete Mathematics*. Addison-Wesley, 1994 (2. kiadás – 1. kiadás 1989). Magyarul: *Konkrét matematika* (szerkesztette Kátai Imre). Műszaki Könyvkiadó, Budapest, 1998. 46, 48

[27] D. E. Knuth. *The Art of Computer Programming. Volume 4. Fascicle 3. Generating of Trees and the History of Combinatorial Generation*. Addison-Wesley, 2006. Magyarul: *A számítógép-programozás művészete. 4. kötet. 3. rész. Fák előállítása. A kombinatorika története*. AnTonCom, Budapest, 2008. 10

[28] R. Krichevski. *Problemy Kibernetiki*, 2:123–138, 1959. 48

[29] R. E. Ladner, M. Fischer. Parallel prefix computation. *Journal of ACM*, 27(4):831–838, 1980. MR0594702 (82a:94115). 32, 33, 44

[30] F. M. Liang. Word hyphenation by computer, PhD disszertáció, 1977. Letölthető a http://www.tug.org/docs/liang/ címről. 10

[31] O. Lupanov. *Izvestiia BUZov, Radiofizika*, 1:120–140, 1958. 16

[32] O. Lupanov. On the complexity of the realization by formulas of the functions of an algebra of logic (oroszul). *Problemy Kibernetiki*, 3:61–80, 1960. MR0126385 (23 #A3681). 48

[33] O. Lupanov. *Doklady Akademii Nauk SSSR*, 140:322–325, 1961. 17, 43

[34] G. Micheli. *Optimization of Digital Circuits*. McGraw-Hill, Cambridge, 1994. 27

[35] R. Muirhead. *Proceedings of the Edinburgh Mathematical Society*, 21:144–157, 1903. 45

[36] E. Necsiporuk. *Doklady Akademii Nauk SSSR*, 163:40–42, 1965. 50

[37] Y. P. Ofman, J. Derbès. *Doklady Akademii Nauk SSSR*, 145:48–51, 1962. 47

[38] E. Paterson. *Giornale di Matematiche*, 25:45–49, 1887. 32, 45

[39] W. Paul. A $2 \cdot 5n$-lower bound on the combinational complexity of Boolean functions. *SIAM Journal on Computing*, 6(3):427–443, 1977. 37, 54

[40] N. J. Pippenger. *Mathematical Systems Theory*, 10:129–167, 1977. MR0489006 (58 #8493). 50

[41] A. Razborov. Lower bounds of monotone complexity of the logical permanent function (oroszul). *Matematicheskie Zametki*, 37(6):887–900 és 942, 1985. MR0802432 (87b:03139). A folyóirat angol fordítása elérhető a http://www.springerlink.com/content/1573-8876/ címen. 30

[42] A. Razborov. Lower bounds on the monotone complexity of some boolean functions (oroszul). *Doklady Akademii Nauk*, 281(4):798–801, 1985. MR0785629 (86k:68032). 30

[43] N. Redkin. A proof of the minimality of certain schemes of functional elements (oroszul). *Problemy Kibernetiki*, 23:83–101, 303, 1970. MR0329799 (48 #8140). 36, 54

[44] N. Redkin. Erratum: A proof of the minimality of certain schemes of functional elements (oroszul). *Problemy Kibernetiki*, 28:4–4, 1974. MR0363733 (50 #16170). 54

[45] N. Redkin. Minimal realization of a binary adder. *Problemy Kibernetiki*, 38:181–216, 272, 1981. MR0640858 (84a:94036). 13

[46] J. Riordan, C. E. Shannon, Jr.http://en.wikipedia.org/wiki/Claude_Shannon. The number of two-terminal series-parallel networks. *Journal of Mathematical and Physical Massachussetts Institute of Technology*, 21:83–93, 1942. MR0007511 (4,151c). 48

[47] J. Savage. *The Complexity of Computing*. Robert E. Krieger Publishing Co., Inc., Melbourne, 1987. MRMR0934574 (89b:68029). A könyvet először a John Wiley & Sons kiadó adta ki 1976-ban. MR0495205 (58 #13936). 47

[48] C.-P.. Schnorr. The combinational complexity of equivalence. *Theoretical Computer Science*, 1(4):289–295, 1975/76. MR0413600 (54 #1714). 37, 55

[49] C. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 59–98. o., 1949. MR0029860 (10,671c). 15, 16

[50] V. Y-S. Shen, A. McKellar, P. Weiner. A fast algorithm for the disjunctive decomposition of switching functions. *IEEE Transactions on Computers*, (3):304–309, 1971. 23, 26

[51] L. B. Sholomov. *Problemy Kibernetiki*, 23:215–226, 1969. 35, 50

[52] J. W. Sklansky. Conditional-sum addition logic. *IRE Transactions on Electronic Computers*, EC-9:505–536, 1960. 46

[53] M. Snir. Depth-size trade-offs for parallel prefix computation. *Journal of Algorithms*, 7:185–201, 1986. MR0106852 (21 #5582). 32, 37, 44, 56

[54] P. M. Spira. *Hawaii International Conference on System Sciences*, 4:525–527, 1971. 30, 39

[55] S. Stein. Two combinatorial covering theorems. *Journal of Combinatorial Theory, Series A*, 16:391–397. oldal, 1974. MR0340062 (49 #4818). 50

[56] L. J. Stockmeyer. The complexity of decision problems in automata theory and logic. Tech. Rep. MAC TR-133, MIT Project MAC, Cambridge, 1974. 28

[57] L. J. Stockmeyer. On the combinational complexity of certain symmetric Boolean functions. *Mathematical Systems Theory*, 10(4):323–336. oldal, 1977. MR0483707 (58 #3687). 37, 55

[58] L. J. Stockmeyer, A. R. Meyer. Cosmological lower bound on the circuit complexity of a small problem in logic. *Journal of ACM*, 49:753–784. oldal, 2002. MR2145856 (2006b:68042). 29

[59] D. Thoreau. *Walden; or, Life in the Woods*. 1854. Az angol változat letölthető a Gutenberg Project honlapjáról: http://www.gutenberg.org/etext/205. 2

[60] D. Uhlig. The synthesis of self-correcting schemes of functional elements with a small number of reliable elements (oroszul). *Matematicheskie Zametki*, 15:937–944, 1974. MR0351655 (50 #4143). 36, 54

[61] D. Uhlig. Networks computing Boolean functions for multiple input values. Boolean function complexity. In *London Math. Soc. Lecture Note Series (Durham, 1990)*, 169. kötet, 165–173. o. Cambridge University Press, 1992. MR1211874 (94a:68057). 54

[62] J. Wakerly. Digital design. 2000. 52

[63] I. Wegener. *The Complexity of Boolean Functions*. Wiley-Teubner Series in Computer Science. John Wiley & Sons and B. G. Teubner, Chichester/Stuttgart, 1987. MR0905473 (89b:03066). 30, 43

[64] A. Weinberger. *IRE Transactions on Electronic Computers*, EC-5:65–73, 1956. 46

[65] F. W. Wood. U.S. Patent 2307868, 1910. november 8. 18

TÁRGYMUTATÓ

NÉVMUTATÓ

MAGYAR NÉVMUTATÓ