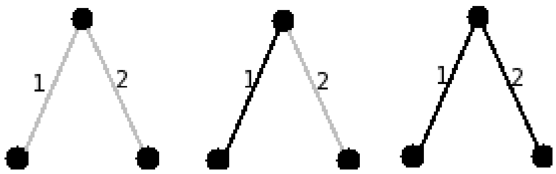
Osztott feszítőfa kereső algoritmus futási ideje átlagos esetben

Sonkoly Pál
phaul@bsd.inf.elte.hu
2006 május, Székesfehérvár

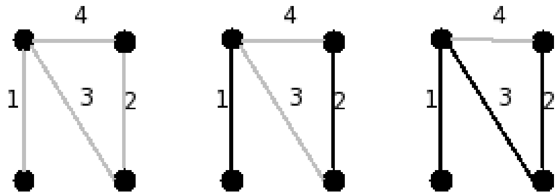
1. Az algoritmus Az algoritmus célja a minimális feszítőfa előállítása összefüggő irányítatlan gráf esetén. Az algoritmus osztott környezetben működik, minden csúcsnak egy-egy folyamat felel meg. Az élek egész értékekkel vannak súlyozva, az általánosság megszorítása nélkül föltehető, hogy ezen értékek különbözőek, s hogy az S = {1, 2,..., |E|} halmazból kerülnek ki (lásd [1]). Az algoritmus megértéséhez szükséges a feszítő erdő és a minimális feszítő erdő fogalma. Az összefüggő gráf feszítő erdeje olyan részgráf, mely az eredeti gráf összes csúcsát tartalmazza, nem feltétlenül összefüggő, de mindenképpen körmentes. A minimális feszítő erdeje egy gráfnak olyan feszítő erdő, mely egy minimális feszítőfájának részgráfja. Világos, hogy $G = (V, E)$ gráf esetén a $G_0 = (V, \emptyset)$ mind feszítő erdő, mind pedig minimális feszítő erdő G-ben. Az algoritmusunk menetekben működik, minden menetben egy minimális feszítő erdőt bővít élekkel, menetenként legalább eggyel. Kezdetben a G_0 triviális minimális feszítő erdőből indulunk ki. A feszítő erdő összefüggő részgráfjait tekintsük egy komponensnek. Egy menetben a komponensek meghatározzák a belőlük kivezető minimális élet (ez az élsúlyok S halmaza miatt egyértelmű). Ezután ezen él mentén elküldik a szomszédos komponensnek, azt az információt, hogy a szóban forgó él a minimális kivezető élük. Ha két komponens minimális kivezető élei megegyeznek, akkor ezt az élet hozzávehetjük az eddigi minimális feszítő erdőhöz, illetve a két komponenst egy komponenssé olvasztjuk össze.

2. Legrosszabb, legjobb eset A futási idő vizsgálatánál most kizárólag a menetek számára korlátozódunk, mindazonáltal a meneteken belüli minimális kivezető él meghatározása vagy a komponensek összeolvasztása is jelentősen befolyásolhatja a futási időt. Legyen $n = |E|$, ekkor az algoritmus legrosszabb esetben $\theta(n\text{-}1)$ menetet hajt végre, hiszen minden menetben legalább egy él hozzáadódik a minimális feszítő erdőhöz (lásd [1]), és a keresett fának pontosan n-1 éle van.



1. ábra

Az első ábrán pontosan ezt az esetet láthatjuk $n = 3$-ra. A szürke élek a gráf azon élei melyek még nem részei az adott menetben a minimális feszítő fának. Az is világos, hogy a legjobb eset akkor áll elő, ha minden menetben a lehető legtöbb élet behúzzuk, vagy másképpen fogalmazva, ha minden komponens egy másik komponenssel összeolvasztásra kerül. Mivel kezdetben n komponens volt, a menetekben a komponensek száma feleződik, ez $\theta(log_2(n))$ menetet igényel.



2. ábra

A második ábra a legjobb esetet mutatja $n = 4$ esetén.

3. Átlagos eset Az átlagos eset futási idejének megadása már nem ilyen egyszerű, a nagyságrend becsléséhez méréseket alkalmaztunk. A mérés lényege, hogy meghatározzuk az összes lehetséges bemenetet adott n-re, majd az ezen bemenetekre produkált menetszámot átlagoljuk. Első lépésben tegyük fel, hogy a gráfunknak $k \in \{1, ..., n*(n-1)/2\}$ éle van. Egészítsük ki a gráfunkat teljes gráffá úgy, hogy az új élekhez extremális súly értéket rendelünk(pl. (-1)-et, hiszen $-1 \notin S$). Az összes lehetséges bemenetet megkaphatjuk a k élű n csúcsú gráfok közül, ha fölírjuk az élsúlyokat egy $n*(n-1)/2$ elemű tömbbe - ennyi éle van a kiegészített gráfunknak -, majd permutáljuk a súlyokat az összes lehetséges módon. A kapott gráfok lehetnek nem összefüggőek is, ezeket nem számoljuk bele az átlagba. Világos, hogy rögzített n-re, az összes lehetséges k-ra a vizsgált gráfok száma

$$\sum_{k=1}^{n} (n! / k!) \approx e\, n!$$

$n = 5$-ig meghatároztuk az össze szóba jöhető esetben a menetszámot.

n	$\#G$	$\sum t$	$\sum t / \#G$
2	1	1	1
3	12	24	2
4	1896	5304	2.7974
5	9854040	35191080	3.5712

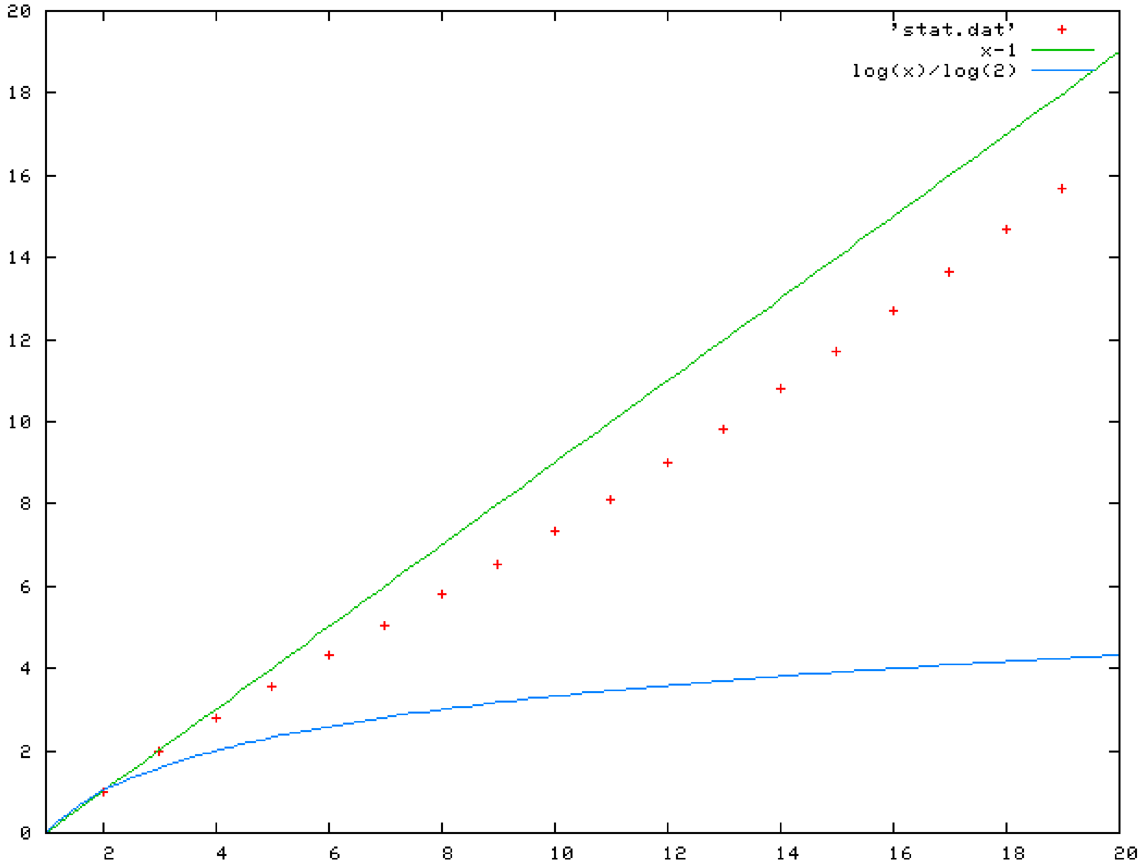
1.táblázat

$\#G$ jelöli adott n-re azon gráfok számát, melyek az előbbiek közül összefüggőek, $\sum t$ a menetszám összegét. $n = 6$ már nem futott le az összegző algoritmus, ehelyett viszonylag nagy és a menetek számát tekintve egyenletesen elszórt mintavétellel dolgoztunk tovább, $n = 19$-ig.

A mintát olyan módon állítottuk elő, hogy egymástól független véletlen gráfokat generáltunk a következő módon. Először pszeudo-véletlen generátorral meghatároztuk az élszámot, így egyforma eséllyel választottunk sűrű és ritka gráfokat. Az élsúlyokat, illetve az élek helyét az összes lehetséges esetből egyenletes eloszlás szerint választottuk.

n	$\#G$	$\sum t$	$\sum t / \#G$
6	1879927502	8093341787	4.3051
7	1353160506	6810685215	5.0331
8	973205083	5643611940	5.7989
9	680373676	4448365879	6.5381
10	28442810	208993174	7.3478
11	19786933	160802129	8.1266
12	13563204	122169712	9.007
13	9264710	91183959	9.8420
14	6408744	69153312	10.790
15	4416038	51635802	11.692
16	3145966	39927891	12.691
17	2193513	29946490	13.652
18	1626029	23872619	14.681
19	1172189	18388696	15.687

2. táblázat



3.ábra

4. Konstans meghatározása A mért adatok alapján feltételezhetjük, hogy várhatóan lineáris idejig fut az algoritmus, a legkisebb négyzetek módszerét alkalmazva meghatározhatjuk a pontokhoz legközelebb eső egyenest. E szerint az algoritmus futási ideje várhatóan $\Theta(0.84*n)+c$.

Irodalomjegyzék

[1] Nancy Ann Lynch Osztott algoritmusok. (Kiskapu kiadó, 2002, ISBN 9639301035)