Alkalmazott Matematikai Lapok **23** (2006), 17–37.

AZ ÁLTALÁNOSÍTOTT LPT(k) ALGORITMUSCSALÁD EGYFORMA PÁRHUZAMOS GÉPEK ÜTEMEZÉSÉRE

DÓSA GYÖRGY ÉS VIZVÁRI BÉLA

Veszprém, Budapest

Egyforma párhuzamos gépek ütemezésével és Graham klasszikus LPT algoritmusának egy újabb általánosításával foglalkozunk. (Korábbi [1] cikkünkben már megadtunk egy másfajta általánosítást.) Az LPT sorrend szerint, egyszerre mindig k számú munkát ütemezünk úgy, hogy a teljes átfutási idő növekedése minimális legyen, vagyis az adott állapot mellett minden lépésben lokálisan optimálisan ütemezzük a soron következő k munkát. Fő eredményünk, hogy minden $2 \leq m \leq 4$ gépszám és minden k érték esetén megadjuk az algoritmuscsalád pontos hatékonyságbecslését, az élességet bizonyító példákkal együtt. Végül tesztfeladatokkal demonstráljuk az algoritmuscsalád gyakorlati hatékonyságát.

1. Bevezetés

Az ütemezéselmélet gyakran vizsgált feladata az egyforma párhuzamos gépek ütemezése, amely a következő: Adott munkák egy $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$ halmaza. Mindegyik munkát m gép valamelyikével kell elvégezni. A T_i munka elvégzésének ideje bármely gépen $l(T_i)$. Ha egy gép egy munkát elkezd elvégezni, akkor azt be is kell fejeznie. A munkák egy ütemezésén $\mathcal{T}$ valamely $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$ partícióját értjük: az i-edik gép végzi a P_i -ben lévő feladatokat. Várakozási idők nincsenek. Az egy gépen elvégzendő munkák sorrendje az ütemezési feladat szempontjából közömbös. A $\mathcal{P}$ ütemezés teljes átfutási ideje

$$\mathcal{L}(\mathcal{P}) = \max_{1 \leq i \leq m} l(P_i), \tag{1}$$

ahol $\mathcal{T}$ tetszőleges X részhalmazára $l(X) = \sum_{T \in X} l(T)$, vagyis $l(P_i)$ az i-edik gépen az utolsó munka befejezési ideje, ahol feltettük, hogy a gépek a 0 időpontban kezdenek dolgozni. A $\mathcal{P}^*$ ütemezés optimális, ha $L(\mathcal{P}^*) \leq L(\mathcal{P})$ a $\mathcal{T}$ halmaz tet-

szőleges $\mathcal{P}$ ütemezése esetén. Optimális ütemezés biztosan létezik, esetleg több is lehet. Az $L(\mathcal{P}^*)$ értéket jelöljük C^*-gal, ami csak $\mathcal{T}$-től és az m számtól függ. A problémával kapcsolatos legfontosabb eredményeket összefoglalja [5].

A feladat az NP-teljes feladatosztályba tartozik, sok heurisztikus módszert dolgoztak ki rá. Ezek között egyik legkorábbi, és egyben a mai napig legnépszerűbb, a Graham-féle LPT (Longest Processing Time) algoritmus [3, 4]. Az algoritmus népszerűségének oka, hogy rendkívül egyszerű, és másrészt sok esetben igen hatékonyan működik. Az algoritmus a következő: Először a munkákat a műveleti idő szerint monoton nemnövekvő sorrendbe rakjuk, majd ebben a sorrendben ütemezzük őket. A következő munka mindig olyan gépre kerül, hogy a lehetséges legkorábbi időpontban fejeződjön be. Az LPT algoritmus vizsgálatával foglalkozó későbbi cikkekről található egy áttekintés korábbi [1] cikkünkben, ahol megvizsgáltuk az algoritmus egy lehetséges általánosítását is. Most egy másik általánosítással foglalkozunk. A jelen cikkben szereplő általánosított algoritmus során (a kezdeti sorbarendezést követően) minden lépésben egyszerre k számú munkát ütemezünk úgy, hogy ezek együttesen a lehető legkevésbé növeljék az átfutási időt. Az algoritmus érdekessége és fő eredményünk, hogy $2 \leq m \leq 4$ gépszám esetén minden k paraméterre pontos elméleti hatékonyság értékeket tudunk majd megadni. Példákkal illusztráljuk majd a módszer gyakorlati hatékonyságát is. Ezek konklúziója, hogy az új algoritmuscsalád sok feladatosztály esetén hatékonyabb az eredeti algoritmusnál.

A cikk szerkezete a következő: A 2. fejezetben megadjuk az algoritmuscsalád pontos definícióját, a 3–6. fejezetekben hatékonyságbecslésekkel foglalkozunk, végül a 7. fejezetben teszteredményeket közlünk.

2. Az általánosított algoritmuscsalád

Jelöljük egy tetszőleges $\mathcal{A}$ algoritmus által a $\mathcal{T}$ feladatra meghatározott ütemezés átfutási idejét $\mathcal{L}_{\mathcal{A}}(\mathcal{T})$-vel. Bevezetjük a következő mennyiséget:

$$R_m(\mathcal{A}) = \sup_{\mathcal{T}} \left\{ \frac{\mathcal{L}_{\mathcal{A}}(\mathcal{T})}{C^*} \right\},$$

és ezt az $\mathcal{A}$ algoritmus elméleti hatékonyságának nevezzük. Ez lényegében azt a szorzót jelenti, ahányszorosa lehet az algoritmus által kapott átfutási idő az optimumértéknek, a lehető legrosszabb esetben. A legrosszabbhoz közeli esetek az összes esetnek csak igen kis százalékában fordulnak elő. Graham LPT algoritmusára igaz a következő

1. TÉTEL. $R_m(\mathrm{LPT}) = \left(\frac{4}{3} - \frac{1}{3m}\right)$. □

A „sup" helyett a képletben „max" is állhat. Legyen ugyanis

$$\mathcal{T}^* = \{2m-1, 2m-1, 2m-2, 2m-2, \ldots, m+1, m+1, m, m, m\}. \tag{2}$$

A halmaz $2m+1$ feladatból áll: kettő-kettő van mindegyik fajtából és a hosszúságuk eggyel csökken, kivéve az utolsó fajtát, amelyből három darab van. Az optimális megoldásnál az utolsó három feladatot egy gépre tesszük, ahol az átfutási idő így $3m$ lesz, a többi feladatot pedig párokba rendezzük úgy, hogy egyet mindig a sor elejéről, egyet pedig mindig a sor végéről veszünk, a két munka hossza együttesen szintén $3m$. Ezzel szemben az LPT algoritmus az első m számú munkát mind különböző gépekre ütemezi, ezek után második munkaként a következő m számú munkát, a gépek fordított sorrendjében, ekkor minden gépen éppen $3m-1$ az átfutási idő. Az utolsó munka elhelyezése után az átfutási idő $4m-1$ lesz. [2] cikkünkben megmutattuk, hogy az előbbi $\mathcal{T}^*$ példán kívül nincs is más olyan feladathalmaz, amelyet LPT ilyen „rosszul" ütemez, minden más feladathalmaz esetén a heurisztikus megoldás értéke kisebb, mint az optimumérték $\left(\frac{4}{3}-\frac{1}{3m}\right)$-szerese.

Az LPT algoritmus a következő munkát a lehető legkorábbi időpontra ütemezi. Helyezzünk el most egyszerre $k \geq 1$ számú munkát úgy, hogy ezek együttesen a lehető legkevésbé növeljék az átfutási időt. $k=1$ esetén nyilván az eredeti LPT algoritmust kapjuk. Az algoritmus formális leírása a következő, ahol m a gépek, n a feladatok száma, $1 \leq k \leq n$ rögzített pozitív egész.

Az általános LPT(k) algoritmus

1. Rakjuk sorba a munkákat csökkenő időtartam szerint. Legyen $n_1 = n$.
2. Legyen $k_1 = \min\{k, n_1\}$.
3. Helyezzük el a soron következő k_1 számú munkát az összes lehetséges módon. Ezek közül válasszuk ki azokat az elhelyezéseket, amikor az átfutási idő növekedése minimális.
4. Az előbbiek közül válasszuk ki azokat az ütemezéseket, amikor a második legnagyobb átfutási idő maximális, ezek közül azokat, amikor a harmadik legnagyobb átfutási idő maximális, és így folytassuk; végül utoljára amikor az $(m-1)$-edik legkisebb átfutási idő maximális (a legkisebb átfutási idő értéke ekkor már adódik).
5. Az így kapott elhelyezések közül válasszunk tetszés szerint, és ütemezzük a kiválasztott módon a munkákat.
6. Legyen $n_1 := n_1 - k_1$. Ha $n_1 > 0$, akkor menjünk a 2. lépésre, egyébként vége.

Figyeljük meg, hogy LPT(k) a soron következő k számú munkát mindig lokálisan optimálisan helyezi el, amin azt értjük, hogy ha nem lenne több munka, csak a soron következő k darab, akkor a pillanatnyi helyzet mellett ennek a hátralévő k darabnak az elhelyezése optimális. Jegyezzük meg, hogy Graham eredeti cikkében is szerepel egy általánosítási lehetőség: A k darab munka optimális ütemezését csak egyszer, az algoritmus elején javasolja, majd a hagyományos LPT-vel folytatja a módosított algoritmust.

2. TÉTEL. *Az* LPT(2) *algoritmus az eredeti LPT algoritmussal azonos ütemezést határoz meg.*

Bizonyítás. Ha egy gépre kerül a soron következő két munka, akkor is; és ha két különböző gépre, akkor is ugyanúgy „helyeződnek el a munkák", mint az LPT algoritmus esetében. □

Definíció. Legyen $\mathcal{A}$ egy ütemezési algoritmus, m a gépek száma. Legyen p, q tetszőleges pozitív egész szám, ahol $p > q$. Egy (p/q) példán olyan $\mathcal{T}$ feladathalmazt értünk, amelyre teljesül, hogy $\mathcal{L}_{\mathcal{A}}(\mathcal{T}) \geq p$, és $C^* = q$. Minimális (p/q) példán olyan $\mathcal{T}$ (p/q) példát értünk, amely minimális az elemszám tekintetében.

1. LEMMA. *Legyen $\mathcal{T}$ egy tetszőleges (p/q) példa. Ekkor tetszőleges $\beta > 0$ esetén, a $\mathcal{T}$-beli munkák mindegyikét β-val megszorozva $(\beta p/\beta q)$ példát kapunk.* □

Az is könnyen látható, hogy az $R_m(\mathcal{A})$ szám, ami azon $\frac{p}{q}$ értékek szupremuma, amelyekhez létezik (p/q) példa, egyben azon $\frac{p}{q}$ értékek infimuma, amelyekhez nem létezik (p/q) példa.

3. TÉTEL. *Tetszőleges rögzített k esetén van olyan m pozitív egész szám, amelyre teljesül a következő egyenlőtlenség: $R_m\big(\mathrm{LPT}(k)\big) \geq \frac{4}{3} - \frac{1}{3m}$.*

Bizonyítás. Az LPT eljárásra $R_m(\mathrm{LPT}) = \frac{4}{3} - \frac{1}{3m} = \frac{4m-1}{3m}$. Tekintsük a (2) példát, ahol $q = 3m$, valamint $p = 4m - 1$. Minden munkából vegyünk k darabot, és k-szorozzuk meg a gépek számát. Ekkor az eredeti LPT algoritmus által adott ütemezéshez hasonlót kapunk, csak az eredetileg szereplő munkák helyén mindegyikből k példány lesz egymás utáni gépekre ütemezve; így a hatékonysági arány is ugyanaz. □

Más a helyzet, ha a gépek m számát rögzítjük, ahogyan ez az alábbi tételből kitűnik:

4. TÉTEL. *Rögzített feladathalmaz esetén $\lim_{k\to\infty} R_m\big(\mathrm{LPT}(k)\big) = 1$, vagyis a k szám növelésével az algoritmus elméleti hatékonysága 1-hez tart.*

Bizonyítás. $k \geq n$ esetén már optimális megoldást kapunk. □

Ezután az egyszerűség kedvéért egy T munka $l(T)$ átfutási idejét többnyire egyszerűen csak T-vel fogjuk jelölni, a legrövidebb ideig tartó munkát A-val jelöljük.

3. Az általános eset

Az alábbiakban minden rögzített m esetén becslést adunk az $\mathrm{LPT}(k)$ elméleti hatékonyságára, és belátjuk, hogy ez k növelésével 1-hez tart. Általában a (2) példa megfelelő módosításai adják majd az élességet bizonyító példákat. Szükségünk lesz a következő lemmára:

2. LEMMA. *Legyen a gépek száma $m \geq 2$ és $\mathcal{T}$ minimális (p/q) példa az $\mathrm{LPT}(k)$ algoritmus esetén. Legyen $A \in \mathcal{T}$ minimális hosszúságú munka. Ekkor $A \geq \frac{m}{m-1}(p - q)$.*

Bizonyítás. A munkák összhossza legföljebb mq. Az egyik gép átfutási ideje az LPT(k) algoritmus által adott ütemezés esetén p. Legyen $\mathcal{T}_1$ azon munkák halmaza, amelyeket az algoritmus utoljára (egyszerre) ütemezett, és ennek az ütemezésnek az ideje legyen a $*$ időpont. Legyen $\mathcal{T}_0 = \mathcal{T}_1 \setminus \{A\}$. Helyezzük el a $*$ időpontban $\mathcal{T}_0$ elemeit úgy, hogy az átfutási idő a lehető legkevésbé növekedjen. Mivel $\mathcal{T} \setminus \{A\}$ nem (p/q) példa $\mathcal{T}$ minimalitása miatt, nem értük el a p átfutási időt. Az ezen munkák ütemezése utáni pillanat legyen a $**$ időpont. Most helyezzük el az A munkát a lehető legkorábbi időpontra. Mivel nem kaphattunk jobb ütemezést, mintha az LPT(k) algoritmus a $\mathcal{T}_1$ elemeit egyszerre helyezte volna el a $*$ időpontban, most is legalább p az átfutási idő, és ez azon a gépen adódik, ahova az A munka került. A többi gép között a legrövidebb átfutási idő legföljebb $q - \frac{p-q}{m-1}$, és ide került az A munka a $**$ időpontban, amikor is elértük a p átfutási időt, ezért A hossza legalább $p - \left(q - \frac{p-q}{m-1}\right) = \frac{m}{m-1}(p-q)$. □

5. TÉTEL. $R_m\big(\mathrm{LPT}(k)\big) \leq \frac{4m-1}{3m}$ *ha* $2 \leq k \leq 2m$. *A becslés éles, ha*

$$2m \equiv 0, 1, 2 \pmod{k}.$$

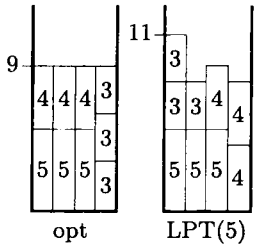
Bizonyítás. Tegyük fel, hogy van olyan $\mathcal{T}$ minimális feladathalmaz, ahol $q = 3m$ és $p > 4m - 1$. A legrövidebb munka hossza a 2. Lemma szerint több, mint m. Emiatt az optimális ütemezésnél minden gépen legföljebb 2 munka van, így a munkák száma legföljebb $2m$. Bármely két munka együttes hossza nagyobb, mint $2m$. Tegyük fel, hogy mindegyik munka hossza kisebb, mint $2m$. Emiatt egyik gépre sem kerül három munka addig, amíg nincs mindegyiken legalább kettő munka. Ekkor az LPT(k) algoritmus ugyanazt az ütemezést eredményezi, mint az LPT algoritmus, és ez optimális megoldás is egyben: A Gantt táblán az alsó sor balról jobbra a felső jobbról balra haladva van feltöltve csökkenő magasságú téglalapokkal. Ha van $2m$-nél hosszabb átfutási idejű munka, az ilyenek az optimális ütemezésnél egyedül vannak a gépeiken. Az LPT(k) algoritmus is csak azután helyezne az ilyen munkák gépeire még egy munkát, amikor a $2m$-nél rövidebb munkák már mind legalább ketten vannak egy-egy gépen, azonban ekkor már nem maradt több munka. Ezzel beláttuk az állítás első részét.

Ameddig az algoritmus alkalmazása során legfeljebb $2m$ munkát ütemeztünk, azok ugyanoda kerülnek, mint ahova őket az LPT algoritmus helyezi. A (2)-beli $\mathcal{T}$ halmaz esetén $2m + 1$ munka van, és ez adja az élességet. Abban az esetben, ha $2m$ k-val osztva 0, 1 vagy 2 maradékot ad, utoljára legfeljebb három munkát helyezünk el egyszerre, ezek egyforma hosszúságúak. Emiatt nem tudjuk őket jobban elhelyezni, mint ha ugyanoda tesszük ezeket is, mint az LPT algoritmus, emiatt a hatékonyság értéke ekkor $\frac{4m-1}{3m}$. □

A becslés éles, ha gépek száma 2 vagy 3, valamint ha $k \leq 4$. Ha a gépek száma $m = 4$, akkor a (2)-beli $\mathcal{T}$ halmazra az LPT(5) algoritmus jobb ütemezést ad, mint az LPT algoritmus. Az alábbi tétel szerint az LPT(5) algoritmus elméleti hatékonysága négy gép esetén valóban jobb, mint $\frac{15}{12}$. Nem végeztünk olyan vizsgálatot, hogy pontosan melyek azok a k számok, amikor $2 < k \leq 2m$ és az előbbi tételben

szereplő becslés nem éles, ugyanis arra fogunk koncentrálni, hogy hogyan változik az elméleti hatékonyság értéke a k paraméter növelésével, a gépek számának rögzítése mellett. Viszont ha a gépek száma legfeljebb négy, és $2 < k \leq 2m$, akkor az előbbi eset az egyetlen kivétel, amikor is LPT (5) hatékonysági szorzója jobb, mint az LPT algoritmusé.

6. TÉTEL. $R_4\big(\mathrm{LPT}(5)\big) = \frac{11}{9}$.



1. ábra

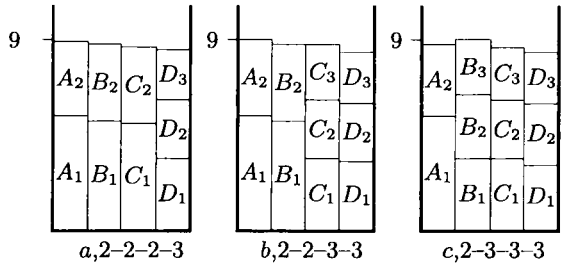
Bizonyítás. Az ábrán látható feladathalmaz (11/9) példa. Tegyük fel, hogy $\mathcal{T}$ minimális $(p/9)$ példa, ahol $p > 11$. A legrövidebb téglalap (a legrövidebb ideig tartó munkának megfelelő téglalap a Gantt táblán) magassága $> \frac{4}{3} \cdot 2 = \frac{8}{3}$, emiatt minden optimális gépen legfőljebb három munka lehet. Ha minden optimális gépen fegfőljebb két munka van, akkor feltehető, hogy azok úgy helyezkednek el, hogy az alul levők hossza balról jobbra, a felül levőké jobbról balra haladva csökken. Nevezzük ezt a sorrendet reguláris sorrendnek. Az első öt munka az algoritmus által ugyanoda kerül, és utána a többi munka is, vagyis az algoritmus a munkákat optimálisan ütemezi. Emiatt van olyan optimális gép, amelyiken három munka van. Ezeken a gépeken a leghosszabb munka hossza is legfeljebb $9 - 2 \cdot \frac{8}{3} = \frac{11}{3}$, a legkisebbnek a hossza pedig legfeljebb 3. Az algoritmus végrehajtása után bármelyik gépre helyezzük át az A munkát, annak teteje a 11 magasságot meghaladja, ezért az A munka nélkül is a gépek átfutási ideje legalább 8, hiszen $A \leq 3$. Emiatt a munkák hosszúságainak összege legalább $11 + 3 \cdot 8 = 35$. Tehát amelyik optimális gépen két munka van, a nagyobbik hossza legalább 4. Belátjuk a következő lemmát:

LEMMA. *Legyen $\mathcal{P}$ a $\mathcal{T} \setminus \{A\}$ olyan ütemezése, amikor a teljes átfutási idő legfeljebb* 11. *Ekkor A elhelyezhető úgy, hogy az ütemezés teljes átfutási ideje továbbra is legfeljebb* 11 *marad.*

Bizonyítás. Legyen a P_i gép átfutási ideje legalább akkora, mint valamelyik optimális gép átfutási ideje. Tegyük az A munkát valamelyik P_i-től különböző gépre. Tegyük fel, hogy az ütemezés átfutási ideje meghaladja a 11-et, ekkor a P_i-től különböző három gép egyike az, ahol több az átfutási idő, mint 11, azaz az optimumértéknél több, mint kettővel nagyobb, a P_i gépen legalább akkora, emiatt a másik két gép átfutási idejénél legalább 2 egységnyi hiány keletkezik, ezért egyiküknek az átfutási ideje kisebb, mint 8. Tegyük akkor ide az A munkát, és nem lépjük túl a

11 egységet, hiszen $A \leq 3$. □

Folytatjuk a tétel bizonyítását. Feltesszük, hogy az optimális gépeken a munkák hossza az ütemezés sorrendjében csökkenő. Vegyük sorra a lehetséges eseteket aszerint, hogy hány optimális gépen van kettő, ill. három munka. Mindegyik esetben azt mutatjuk meg, hogy az LPT(5) algoritmus által adott teljes átfutási idő legfeljebb 11.



2. ábra

a) 2-2-2-3 munka van az optimális gépeken. Feltehető, hogy az első három gépen a munkák sorrendje reguláris. A három leghosszabb idejű munka az A_1, B_1 és a C_1. Helyezzük el az első öt munkát, ekkor a negyedik gépen kettő munka lesz. Az ütemezetlen munkák között szerepel az A_2 és a D_3 munka, mert mindkettő esetén van másik öt olyan munka, amelyeknek a hossza legalább akkora. D_3 hossza legfeljebb 3. Tegyük az A_2 munkát az első gépre, itt ugyanazok a munkák vannak most, mint az optimális ütemezésnél, emiatt ugyanakkora a gép átfutási ideje. A másik három munka közül a D_3-tól különböző két munka elhelyezhető most a második és harmadik gépen úgy, hogy ne lépjük túl a 11 időt, (sőt 9-et sem), mert most ide legfeljebb olyan hosszúságú munkák kerülnek, mint amilyenek az optimális ütemezésnél voltak. Most alkalmazzuk a lemmát, és kapunk egy legfeljebb 11 átfutási idővel rendelkező ütemezést.

b) 2-2-3-3 munka van az optimális gépeken. Feltehető, hogy az első két gépen a munkák sorrendje reguláris. A két leghosszabb idejű munka A_1 és B_1. Az utolsó két gépen levő munkák hossza kisebb, mint 4, emiatt az öt legkisebb mindegyikének a hossza is kisebb, mint 4. Helyezzük el az első öt munkát. Tegyük fel, hogy ezek között az A_2 munka nem szerepel. Ha az első öt munka között ott van C_3, akkor C_2 és C_1 is, D_3 pedig nincs. Ugyanígy, ha D_3 ott van, akkor C_3 nincs. Emiatt a maradék öt munka közül az egyik az A_2, a másik négy hossza kisebb, mint 4, és van olyan is e négy között, amelyiknek a hossza legfeljebb 3.

Tegyük az A_2 munkát az első gépre, ekkor a gép ütemezése nem változott. A másik négy munka közül a három nagyobbikat próbáljuk úgy elhelyezni a másik három gépen, hogy egyiknek az átfutási ideje se lépje túl a 11-et. Ezeken a gépeken most csak olyan munka van, mint az optimális ütemezésnél, és hiányzik még egy legalább $\frac{8}{3}$ hosszúságú munka. Ha valamely gép átfutási ideje több lenne, mint 11, akkor a másik két gép közül van olyan, amelynek az átfutási ideje kisebb, mint

$27 - 11 - \frac{8}{3}$ fele, emiatt kisebb, mint 7. Az előbb említett három nagyobbik munka mindegyikének a hossza kisebb, mint 4. Ha tehát valamelyik gép átfutási ideje túllépné a 11-et, akkor a túlnyúló munkát tegyük a 7-nél kisebb átfutási idejű gépre. Ekkor egyik gép átfutási ideje sem több, mint 11. Alkalmazzuk a lemmát, és kapunk egy 11-nél nem nagyobb idejű ütemezést.

Hátramaradt az az eset, hogy A_2 az első öt munka között van. Ez csak úgy lehet, hogy az öt leghosszabb munka: A_1, A_2, B_1 és B_2, valamint a másik két optimális gép egyikéről a leghosszabb munka, például C_1. Mivel A_1 és B_1 a két leghosszabb munka, ezek az első öt elhelyezésekor külön gépre kerülnek. A másik három munka két gépre kerül. Legyen közülük X az, amelyik öt közül egyedül marad. Tegyük még erre a gépre a D_2 és D_3 munkákat. Ennek a gépnek az átfutási ideje most legalább $D_1 + D_2 + D_3$, mert $X \geq D_1$. Ha $X + D_2 + D_3 > 11$ lenne, akkor $X > D_1 + 2 \geq 4\frac{2}{3}$, ami nem lehet, hiszen B_1, B_2 és C_1 mindegyike ennél kisebb. Maradt még három munka. Tegyük a két nagyobbat az első két gépre, ezek legfeljebb akkorák, mint az optimális ütemezéskori kisebbik munkák, így egyik gép átfutási ideje sem több, mint 11, és a lemma alkalmazható.

c) 2-3-3-3 munka van az optimális gépeken. Ha A_1 és A_2 egy gépre került az első tíz munka elhelyezésekor, tegyük olyan másik gépre az utolsó munkát, amelyen csak két munka van. E három munka együttes hossza legfeljebb $3 \cdot \frac{11}{3} = 11$. Ha A_2 marad utoljára, akkor ha A_1 gépén nincs más munka, oda befér. Ha van, akkor van olyan másik gép, ahol csak két munka van, és ide elfér, mert e három munka mindegyikének a hossza legfeljebb $\frac{11}{3}$. Ha az A_1 és A_2 két különböző gépre került az első tíz munka elhelyezésekor, akkor ha a másik két gép közül valamelyiken csak két munka van, ide elfér az utolsó munka. Ha viszont e másik két gépen három-három munka van, az A_1 és A_2 gépén pedig még egy-egy munka, legyen az A_2 gépén például még rajta kívül az X munka. Legyen az Y olyan munka, amelyik ugyanazon az optimális gépen van, mint az X. Ekkor $Y + X + A \leq 9$. Ha $A_2 + X + A > 11$, akkor $A_2 > Y + 2$, emiatt $A_2 > 4\frac{2}{3}$, ami ellentmond annak, hogy A_2 az első optimális gépen a kisebbik munka. Eszerint az utolsó munka elfér még azon a gépen, amelyiken A_2 van, és az átfutási idő nem lesz nagyobb, mint 11.

d) 3-3-3-3 munka van az optimális gépeken. Mivel a legkisebb munka hossza több, mint $\frac{8}{3}$, a leghosszabb munka hossza is kevesebb, mint $\frac{11}{3}$. Helyezzük el az első öt munkát, majd a következő ötöt. Mivel minimális példáról van szó, egyik gép átfutási ideje sem lépi túl a 11-et. Van két olyan gép, amelyeken legfeljebb csak két munka van. Tegyük ezekre az utolsó két munkát. Mivel három munka átfutási ideje együtt kisebb, mint 11, ellentmondást kaptunk. □

Az 5. Tétel szerint $R_m(\mathrm{LPT}(2m)) = \frac{4m-1}{3m}$. Ezt általánosítja az alábbi tétel:

7. TÉTEL. $R_m(\mathrm{LPT}(\gamma m)) = \frac{2m-1+\gamma m}{m+\gamma m}$, *ahol* $\gamma \geq 2$, *tetszőleges pozitív egész.*

Bizonyítás. A becslés értéke legalább ennyi, ugyanis a (2) példát bővítsük ki $(\gamma - 2)m$ számú m méretű munkával, és megkapjuk az előbbi értéket. Másrészt tegyük fel, hogy van olyan $\mathcal{T}$ minimális feladathalmaz, ahol az optimális megoldás értéke $m + \gamma m$, a heurisztikus megoldás értéke pedig több, mint $2m - 1 + \gamma m$.

A legrövidebb munka hossza több, mint m. Emiatt minden optimális gépen legföljebb γ munka van, így a munkák együttes száma legföljebb γm, ezekre viszont algoritmusunk optimális megoldást ad, ellenmondást kaptunk. □

KÖVETKEZMÉNY. $R_m(\mathrm{LPT}(\gamma m + l)) \leq \frac{2m-1+\gamma m}{m+\gamma m}$, *ahol* $\gamma \geq 2$, *tetszőleges pozitív egész, valamint* $0 \leq l < m$ *nemnegatív egész szám.*

Bizonyítás. A munkák száma az előbbi módon kiszámítva megint legföljebb γm, ezekre viszont a heurisztikus algoritmusunk optimális megoldást ad, ellenmondást kaptunk. □

Ha k értéke nagy, akkor a becslést egy kicsit javítani tudjuk a következő három tétel szerint.

8. TÉTEL. $R_m(\mathrm{LPT}(\gamma m + 1)) \leq \frac{2m+\gamma m}{m+\gamma m+1}$, *ahol* $\gamma \geq 2$, *valamint* $\gamma \geq m - 3$.

Bizonyítás. Tegyük fel, hogy van olyan $\mathcal{T}$ feladathalmaz, ahol $q = m + \gamma m + 1$, míg $p > 2m + \gamma m$. A legrövidebb A téglalap magasságára $A > m$. Emiatt az optimális ütemezésnél minden gépen legföljebb $\gamma + 1$ munka van, ami összesen legföljebb $\gamma m + m$ munka. Emiatt pontosan két ütemben helyezzük el a munkákat. A két ütem közti időpont $*$. A téglalapok összterülete legföljebb $m(m + \gamma m + 1)$. A $*$ időpontban még nem értük el a $2m + \gamma m$ magasságot, különben a többi munka elhagyható lenne.

A $*$ időpontban egyik gépen sem lehet $\gamma + 2$ vagy több munka, mert az ezeknek megfelelő téglalapok együttes hossza meghaladná a $\gamma m + 2m$ magasságot. Ezért mindegyik gépen legfeljebb $\gamma + 1$ munka van, és van ahol éppen ennyi, mert már $\gamma m + 1$ munkát elhelyeztünk. Tekintsünk egy ilyen gépet. Az első γ munka együttes ideje több, mint γm. Legalább ekkora az átfutási idő a többi gépen, különben a $(\gamma + 1)$-edik munkát nem ide tettük volna. Ezért a második ütemben egyik gépre sem kerülhet kettő vagy több munka, ugyanis a $*$-kor meglevő több, mint γm átfutási időhöz így még hozzá jön több, mint m (az egyik munka ideje), ez már több, mint $\gamma m + m$, ekkora átfutási idő mindegyik gépen van a végén, mert akkor a másik munka nem erre a gépre kerülne, ez összesen több, mint $m(m + \gamma m)$, és még ehhez hozzájön a másik munka ideje, ami így összesen meghaladja a téglalapok összterületét. Ugyanígy látható be, hogy az eljárás végén egyik gépen sem lehet $\gamma + 2$ vagy több munka. Tehát $*$ időpontban néhány gépen pontosan $\gamma + 1$ számú munka van, a többi gépen ennél kevesebb. Az algoritmus második lépésében a maradék munkák csak olyan gépekre kerülnek, ahol még legfeljebb γ munka van, továbbá ezek a $*$ időpont után elhelyezett munkák mind különböző gépre kerülnek. Ebből következik, hogy a maradék munkákat az eredeti LPT szabály szerint rakjuk le. Így a minimalitás miatt feltehető, hogy csak egyetlen feladatnak, és éppen egy minimális idejűnek az átfutási ideje fogja csak meghaladni az algoritmus végén a $\gamma m + 2m$ értéket.

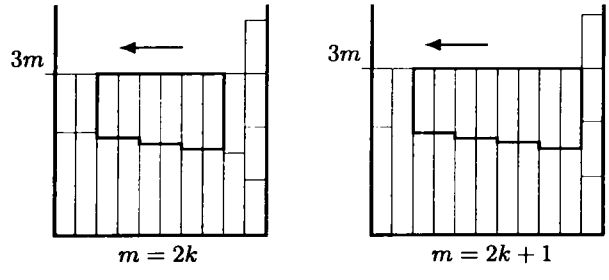
Legyen $A = m + \varepsilon$, ahol $\varepsilon > 0$. Az eljárás végén egy olyan gép van, ahol az átfutási idő több, mint $\gamma m + 2m$, ami az optimumértéknél több, mint $(m - 1)$-gyel több. A többi gépen az átfutási idők átlaga kisebb, mint $m + \gamma m$, hiszen az átfutási idők

összege a téglalapok összterületét nem haladhatja meg. A $*$ időpontban amelyik gépen pontosan $\gamma+1$ munka van, ott az átfutási idő több, mint $\gamma m+m+(\gamma+1)\varepsilon$. Legalább egy ilyen gép van, ezen gép(ek)en az előbbi átlagmagassághoz képest legalább $(\gamma+1)\varepsilon$-nyi többlet keletkezik. Az algoritmus végén lesz legalább egy olyan gép, ahol az átfutási idő nem éri el a $m+\gamma m$ értéket, de az ilyen gépek száma legföljebb $m-2$. Mivel az eredeti feltevés szerint teljesül $\gamma \geq m-3$, ezért $\gamma+1 \geq m-2$, így az algoritmus végén lesz olyan gép, amelyiknek az átfutási ideje kisebb, mint $m+\gamma m-\varepsilon$. Ellentmondáshoz jutottunk, mert akkor az A munkát erre a gépre helyezve a heurisztikus megoldás értéke csökkenne. □

Az alábbi példa mutatja, hogy szükséges a $\gamma \geq m-3$ feltétel. Legyen $m=6$, $\gamma=2$ és $\mathcal{T}=\{9,9,8,8,7,7,6,6,5,5,5,5,5,5\}$. Ekkor a munkák optimális elhelyezésekor mindegyik gép átfutási ideje 15 lesz: $P_1^*=P_2^*=\{9,6\}$, $P_3^*=P_4^*=\{8,7\}$, $P_5^*=P_6^*=\{5,5,5\}$. Az LPT(13) algoritmus először elhelyez 13 munkát a következőképpen: $P_1=P_2=\{9,5\}$, $P_3=P_4=\{8,6\}$, $P_5=\{7,7\}$, $P_6=\{5,5,5\}$. Ekkor az első öt gép átfutási ideje 14, az utolsóé 15. Az utolsó munka ütemezése után az átfutási idő 19 lesz. Mivel $\frac{19}{15}>\frac{24}{19}$, emiatt az előbbi tételben szereplő becslés hat gép és $\gamma=2$ esetén már nem teljesül. Ekkor így a hatékonyság értéke legalább $\frac{19}{15}$.

9. TÉTEL. $R_m\big(\mathrm{LPT}(\gamma m+1)\big) \geq \frac{2m+\gamma m}{m+\gamma m+1}$, *ahol* $\gamma \geq 2$.

Megjegyzés. Most a $\gamma \geq m-3$ feltételre nincs szükség. Állításunkból következik, hogy az $\mathrm{LPT}(\gamma m+1)$ algoritmus hatékonyságbecslése $\gamma \geq \max\{2, m-3\}$ esetén éles.



3. ábra

Bizonyítás. Legyen először $\gamma=2$. Megmutatjuk, hogy ha a (2) példához még egy darab m hosszú téglalapot hozzáveszünk, akkor a hatékonyság értéke $\frac{2m+\gamma m}{m+\gamma m+1}=\frac{4m}{3m+1}$. Az ábrákon a (2) feladathalmaz optimális elhelyezése látható, valamint a hozzávett még egy téglalap az előbbiek fölött. Az első két gépen levő munkák ideje $2m-1$ és $m+1$, a következő két gépen levő munkáké $2m-2$ és $m+2$, és így tovább. Ha a gépek száma páros: $m=2k$, akkor az utolsó előtti gépen levő mindkét munka ideje $3k$; ha $m=2k+1$, akkor az utolsó előtti és azelőtti gépen levő munkák ideje $3k+2$ és $3k+1$. Az utolsó gépen mindkét esetben négy darab m hosszúságú

munka van. Az LPT($2m+1$) algoritmus az előbbi elhelyezéseket produkálja. Az optimális elhelyezéseket a következőképpen kapjuk: A rövidebb ideig tartó munkákat kettő géppel balra tesszük, az első esetben ez a 3.-tól az $(m-2)$.-ig terjedő munkákra, a másik esetben szintén a 3.-tól az $(m-1)$.-ig terjedő munkákra vonatkozik, az ábrán ezek a vastag vonallal keretezett részek. Az érintett gépeken az átfutási idő megnövekszik 1-gyel. Ha a gépek száma páros, akkor tegyük az utolsó előtti gépen levő két munkát az előtte levő két gépre. Ekkor kimaradt hat munka: az első két gépről származik két $m+1$ hosszúságú, és az utolsóról négy m hosszúságú munka. Helyezzük el ezeket a maradék két gépre egyformán. Ekkor az átfutási idő mindegyik gépen $3m+1$ lett. Ha a $m=2k+1$, akkor még az utolsó három gép ütemezése maradt hátra, és nyolc munkát nem helyeztünk még el: Az első két gépről származik két $m+1$ hosszúságú munka, az utolsó előtti és azelőtti gépről két $3k+2$ hosszúságú munka, valamint az utolsóról négy m hosszúságú munka. Tegyük a $3k+2$ hosszúságú munkákat egy gépre, a többit pedig osszuk el a maradék két gép között egyformán. Az átfutási idő most is mindegyik gépen $3m+1$ lett. Így kaptunk egy $\frac{4m}{3m+1}$ hatékonyságú példát. Tetszőleges $\gamma \geq 2$ esetén a szokásos módon kapjuk a megfelelő példát: Adjunk a feladathalmazhoz még $(\gamma-2)\cdot m$ számú m méretű munkát. □

KÖVETKEZMÉNY. $R_m(\mathrm{LPT}(\gamma m+1)) = \frac{2m+\gamma m}{m+\gamma m+1}$, *ahol* $\gamma \geq 2$ *és* $\gamma \geq m-3$ *(vagyis például* $m \leq 5$ *esetén a 8. Tétel becslése éles).*

10. TÉTEL. $R_m(\mathrm{LPT}(\gamma m+l)) \leq \frac{2m+\gamma m}{m+\gamma m+1}$, *ahol* $\gamma \geq m$ *tetszőleges pozitív egész, valamint* $1 \leq l < m$ *nemnegatív egész szám.*

Bizonyítás. Indirekt módon tegyük fel, hogy $\mathcal{T}$ olyan minimális feladathalmaz, amelyre $\mathrm{LPT}(\gamma m+l) > 2m+\gamma m$, de $C^* = m+\gamma m+1$. Válasszunk ezek között a feladathalmazok között olyat, amikor az l szám minimális. Mivel $l=1$ esetén az állítás adódik az előző tételből, $l>1$. A 2. Lemma szerint a legrövidebb téglalap magassága $A > m$. Így optimális ütemezésnél minden gépen legföljebb $\gamma+1$ munka van, tehát a munkák száma legföljebb $m+\gamma m$. Mivel $\gamma m+l$ vagy ennél kevesebb számú feladat esetén optimális megoldást kapnánk, a feladatok száma ennél több. A $*$ időpontban összesen $\gamma m+l$ számú munkát helyeztünk már el a gépekre, a többi munkát már mind elhelyezi egyszerre az algoritmus a következő lépésben. $*$-kor van olyan gép, amelyiken γ -nál több munka van, vagyis a $\gamma \geq m$ feltétel miatt a $*$ időpontban van olyan gép, amelyiken m-nél több munka van. Ezek között van két olyan, amelyek az optimális ütemezés esetén azonos gépeken vannak. Módosítsuk úgy a feladathalmazt, hogy e két munka helyett vegyünk egyetlen olyat, aminek a végrehajtási ideje egyenlő az előbbi két munka idejének összegével. Legyen ez a $\mathcal{T}'$ feladathalmaz. Most végezzük el az $\mathrm{LPT}(\gamma m+l-1)$ algoritmust a $\mathcal{T}'$ feladathalmazzal. A $*$ időpontban minden gép átfutási ideje ugyanannyi lesz, mint az előbb, és a maradék téglalapok ugyanoda kerülnek, mint ahova $\mathrm{LPT}(\gamma m+l)$ helyezte az előbb a $\mathcal{T}$ maradék téglalapjait. (Most használtuk fel először az algoritmus 4. pont-

jának teljesülését!) Így most is ugyanolyan hatékonyságú példát kaptunk, mint az előbb, ez pedig ellentmond az l szám minimális választásának. □

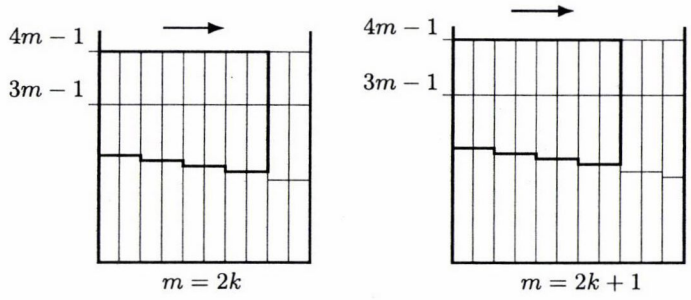
Az eddigieket összefoglalva elmondhatjuk, hogy minden m-re és k-ra van eléggé pontos felső becslésünk, ami ráadásul éles is, ha $2 \leq k \leq 2m$, és $2m$ k-val osztva legfeljebb kettő maradékot ad, illetve ha $k \geq \max\{2m, (m-3)\,m\}$, és k m-mel osztva 0 vagy 1 maradékot ad.

Ha $m = 2$, akkor az előző tételeket alkalmazva minden k-ra pontos becslést tudunk megadni. Ha a gépek száma három, akkor a „kicsi" k értékek 2 és 6 közöttiek, a „nagyok" pedig 9-től kezdődnek, ezek mindegyikére pontos becslésünk van. $k = 7$ esetére alkalmazható a 10. Tétel előtti következmény, csak $k = 8$ esete maradt ki, erre a következő tételben fogunk pontos becslést megadni. Ha a gépek száma négy, és a k értéke legföljebb 8 vagy legalább 16, akkor alkalmazhatók az előbbi eredmények. Az előbbi határok között tudjuk még az előző tételek alapján a pontos hatékonysági értéket 9, 12, és 13 esetén, a többi k-ra az elméleti hatékonyság értéke szabálytalan. A következő tételben megvizsgáljuk a $k = 3m - 1$ esetet. A következő fejezetekben 2, 3 és 4 gép esetén *minden* k-ra meghatározzuk az elméleti hatékonyság pontos értékét.

11. TÉTEL. $R_m(\mathrm{LPT}(3m-1)) = \frac{5m-2}{4m-1}$.

Bizonyítás. Tegyük fel, hogy $\mathcal{T}$ olyan feladathalmaz, ahol $q = 4m - 1$, míg $p > 5m - 2$. Ekkor $A > m$. Emiatt optimális gépen legföljebb 3 munka van. Mivel $3m$-nél kevesebb munka esetén az eljárás optimális megoldást ad, pontosan $3m$ a munkák száma, és minden optimális gépen három munka van. Tegyük fel, hogy a $*$ időpontban van olyan gép, amelyre két munkát, C_1-et és C_2-t ($C_1 \geq C_2$) osztottunk. Az utolsó munkát erre a gépre téve az átfutási idő meghaladja az $5m - 2$ értéket, ugyanis ha a legkorábbi időpontra ütemezzük az algoritmusunk szerint, akkor is meghaladja ezt az értéket. Emiatt teljesül a $C_1 + C_2 + A > 5m - 2$ egyenlőtlenség, ahonnan $2C_1 + A > 5m - 2$. Mivel a C_1 munka mellett optimális ütemezésnél még két másik munka van ugyanazon a gépen, teljesül a $C_1 + 2A \leq 4m - 1$ egyenlőtlenség is. A két egyenlőtlenséget összevetve $A < m$ adódik, ami ellentmondás. Ellentmondásra jutunk akkor is, ha a $*$ időpontban van olyan gép, amelyiken csak egy munka van: Jelöljük ezt az egy munkát C-vel. Ekkor $C + A > 5m - 2$, de akkor optimális ütemezésnél is egyedül kellene a C munkának lennie. Az előzőek alapján a $*$ időpontban mindegyik géphez legalább három munka lett már rendelve, ami ellentmond annak, hogy eddig a pillanatig még csak $3m - 1$ munkát osztottunk szét a gépek között.

A becslés éles voltát a (2) példa azon módosításával látjuk be, ahol (2)-t kiegészítettük $m - 1$ darab m hosszú téglalappal. Ábránkon (2) LPT algoritmus szerinti elhelyezése látható, kiegészítve a hozzávett téglalapokkal. Minden gépen három munka van, a legrövidebb munka ideje mindig m. Az első két gépen levő másik két munka ideje $2m - 1$ és m, a következő két gépen pedig $2m - 2$ és $m + 1$, és így tovább. Ha a gépek száma páros: $m = 2k$, akkor az utolsó előtti két gépen levő további munkák időtartama $3k$ és $3k - 1$; ellenkező esetben az utolsó gépen levő



4. ábra

mindkét további munka időtartama $3k+1$. A heurisztikus megoldásban a vastag vonallal keretezett téglalapok két géppel jobbra kerülnek, ahol az átfutási idő 1-gyel csökken.

Ha a gépek száma páros, akkor marad még két gép (az első és a második), valamint hat munka: az első két gépről származik még egy-egy $2m-1$ hosszúságú, és az utolsó kettőről két $3k-1$ és két m hosszúságú munka. Az egyik gépre kerül a két $2m-1$ hosszúságú, a másikra két $3k-1$ és egy m hosszúságú munka. Ezek a munkák helyeződnek el az LPT$(3m-1)$ algoritmus első lépésében, az átfutási idő ekkor minden gépen pontosan $4m-2$, és még hátramaradt egyetlen m idejű munka, ezeket elhelyezve lesz az algoritmus átfutási ideje egyik gépen $5m-2$.

Ha a gépek száma páratlan, akkor még három gép maradt, az első kettő és az utolsó, valamint kimaradt még kilenc munka. Az első két gépről származó két $2m-1$ hosszú munkát tegyük egy gépre. Hátra van még kettő $3k$, kettő $3k+1$ és három m hosszúságú munka. Helyezzünk el két gépre három-három munkát a következőképpen: egy $3k$, egy $3k+1$, és egy m hosszúságút. Most mindegyik gépen pontosan $4m-2$ az átfutási idő, és ezt az elhelyezést valósítja meg az LPT$(3m-1)$ algoritmus az első lépésében. Megint maradt még egyetlen m idejű munka, ezt elhelyezve lesz az algoritmus átfutási ideje éppen $5m-2$. □

12. TÉTEL. $R_m\big(\mathrm{LPT}(k)\big) < \frac{m+k}{k+1}$ *ha* $k \geq 2m$.

Bizonyítás. Legyen $k=\gamma m+l$, ahol $0 \leq l < m$. Teljesül a következő egyenlőtlenség-lánc: $R_m\big(\mathrm{LPT}(\gamma m+l)\big) \leq \frac{2m+\gamma m}{m+\gamma m+1} < \frac{m+\gamma m+l}{\gamma m+l+1} = \frac{m+k}{k+1}$. □

KÖVETKEZMÉNY. *Az előbbi tétel szerint rögzített m esetén az algoritmusunk elméleti hatékonysága 1-hez tart:* $\lim_{k\to\infty} R_m\big(\mathrm{LPT}(k)\big) = 1$.

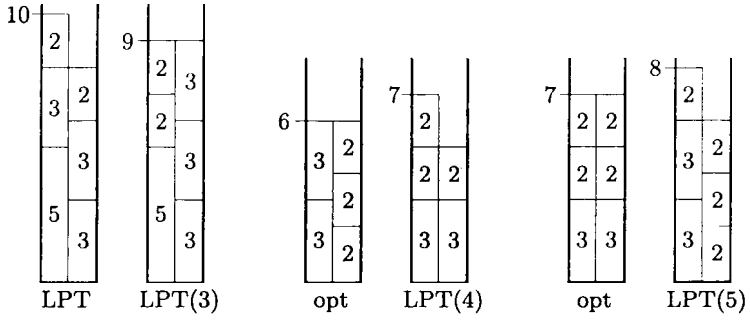
A következő három fejezetben megvizsgáljuk két, három, illetve négy gép esetén az összes lehetséges esetet.

4. $m = 2$ esete

Legyen $m = 2$, $\mathcal{T} = \{5, 3, 3, 3, 2, 2\}$. Az 5. ábra bal oldali része mutatja, hogy az LPT algoritmusnak 10, míg LPT(3)-nak 9 egység a teljes átfutási ideje.

Az 5. Tétel szerint az LPT(3) és LPT(4) algoritmusok hatékonysága $R_2 = \frac{7}{6}$: Álljon $\mathcal{T}$ a következő téglalapokból: $\mathcal{T} = \{3, 3, 2, 2, 2\}$. Ekkor az LPT(3) és LPT(4) által meghatározott átfutási idő 7, míg az optimális megoldás értéke 6 egység (5. ábra). A legelső k, amelyre javul algoritmusunk hatékonysága, a $k = 5$. Az előző fejezetbeli 6., 7. és 8. Tételek alapján tetszőleges k-ra megadtuk az algoritmusunk hatékonyságát. $m = 2$ esetén azonban az algoritmus hatékonyságát közvetlenül és egyszerre, tetszőleges $k \geq 2$ esetére is meg tudjuk határozni, az alábbi tétel szerint:

13. TÉTEL. *Ha* $k \geq 4$, *akkor* $R_2\big(\mathrm{LPT}(k)\big) = \frac{k+3}{k+2}$.

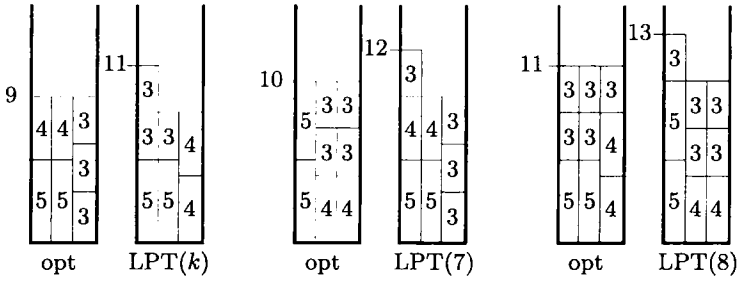


5. ábra

Bizonyítás. $\frac{k+3}{k+2}$-es példának megfelel a következő: Álljon a $\mathcal{T}$ feladathalmaz 2 darab 3 egység magasságú, és $k - 1$ darab 2 egység magasságú téglalapból. Ekkor az LPT(k) által meghatározott átfutási idő $k + 3$, míg az optimális megoldás értéke $k + 2$. (Az optimális és heurisztikus megoldások ugyanis $k = 4$ és $k = 5$ esetén az 5. ábrán szerepelnek. Nagyobb k számok esetén pedig az előbbi téglalapok fölé kerül még néhány sor a 2 hosszúságú téglalapból, páros k esetén a középső, páratlan k esetén pedig a jobb oldali ábrát véve alapul.) Meg kell mutatnunk még, hogy $R_2\big(\mathrm{LPT}(k)\big) \leq \frac{k+3}{k+2}$. Tegyük fel ezzel ellentétben, hogy létezik olyan $\mathcal{T}$ feladathalmaz, ahol az optimális megoldás értéke éppen $k + 2$, a heurisztikus megoldás értéke több, mint $k + 3$. A téglalapok összterülete legföljebb $2 \cdot (k + 2)$. A 2. Lemma miatt a legrövidebb téglalap magassága több, mint 2. Emiatt $k + 2$-nál kevesebb téglalap van, $k + 1$-nél kevesebb nem lehet, mert ekkor LPT(k) optimális megoldást határozna meg, így pontosan $k + 1$ darab téglalap van. Az első k számú téglalapot LPT(k) ezekre vonatkozóan optimálisan helyezi el, ezek $\mathcal{T}$ minimalitása miatt beleférnek $k + 3$ sávmagasságba. Az utolsó téglalap az alacsonyabb helyre kerül, ekkor növekszik a sávmagasság $k + 3$ fölé. A másik magasság ezért kisebb, mint $k + 1$, az utolsó téglalap elhelyezése előtt ezért mindkettőé kisebb, mint $k + 1$. Legyen k páratlan.

Ekkor az első k darab téglalap elhelyezésekor az egyik gépre legalább $\frac{k+1}{2}$ feladat került, ezek együttes átfutási ideje több, mint $k+1$, ellentmondást kaptunk. Most legyen az k páros szám. Mivel összesen $k+1$ számú téglalap van, az optimális megoldásnál van olyan gép, amelyikre legalább $\frac{k+2}{2}$ számú munka kerül, ezek együttes átfutási ideje több, mint $k+2$, megint ellentmondást kaptunk. □

5. $m = 3$ esete



6. ábra

Ha $m = 3$, az LPT algoritmus éles felső becslése $\frac{11}{9}$. A $\mathcal{T} = \{5,5,4,4,3,3,3\}$ feladathalmaznál az optimum értéke 9, míg az LPT által adott ütemezésé 11. Az LPT(k) algoritmus ugyanezt az elhelyezést valósítja meg $3 \le k \le 6$ esetén, és az 5. Tétel szerint $R_3\big(\mathrm{LPT}(k)\big) = \frac{11}{9}$ $(3 \le k \le 6)$. Továbbá a 6. Tételben beláttuk, hogy $R_3\big(\mathrm{LPT}(6+3\gamma)\big) = \frac{11+3\gamma}{9+3\gamma}$. A 7. és 8. Tételek szerint $R_3\big(\mathrm{LPT}(7+3\gamma)\big) = \frac{12+3\gamma}{10+3\gamma}$, ahol $\gamma \ge 0$. A becslés éles voltát bizonyító példát úgy kaptuk, hogy a (2) példához hozzávettünk még egy legrövidebb idejű munkát, az optimális, illetve heurisztikus megoldásokat mutatja $\gamma = 0$ esetén a 6. ábra. Nagyobb γ esetén a szokásos módon járunk el: hozzáveszünk a feladathalmazhoz 3γ számú, 3 egység időtartamú munkát.

A 11. Tétel szerint $R_3\big(\mathrm{LPT}(8)\big) = \frac{13}{11}$. Az éles példa $\mathcal{T} = \{5,5,4,4,3,3,3,3,3\}$. Három gép esetén az LPT$(11+3\gamma)$ algoritmus elméleti hatékonysága legfeljebb $\frac{15+3\gamma}{13+3\gamma}$, ha $\gamma \ge 0$, a 10. Tétel szerint. Most megmutatjuk, hogy ez a becslés éles.

14. TÉTEL. *Ha* $\gamma \ge 0$, *akkor* $R_3\big(\mathrm{LPT}(11+3\gamma)\big) = \frac{15+3\gamma}{13+3\gamma}$.

Bizonyítás. Csak azt kell belátnunk, hogy a hatékonyság értéke legalább ekkora. Legyen először $\gamma = 0$. Tekintsük a $\mathcal{T} = \{4,4,4,3,3,3,3,3,3,3,3,3\}$ feladathalmazt. Az optimum értéke 13: mindegyik optimális gépen egy darab 4, és három darab 3 hosszúságú munka van. Az LPT(11) által kapott megoldás átfutási ideje 15: az első gépre kerülnek a 4 hosszúságú munkák, a másik két gépre négy-négy darab 3

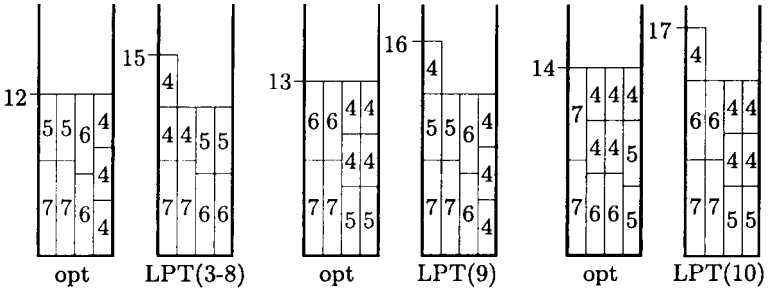
hosszúságú munka, az algoritmus első ütemében. Ekkor minden gép átfutási ideje 12, és még hátravan egy darab 3 egység hosszú munka elhelyezése. Nagyobb γ számok esetén elég az előbbi $\mathcal{T}$ halmazhoz még 3γ darab 3 egység hosszúságú téglalapot kell hozzávenni, ekkor az optimális és a heurisztikus megoldás értéke is 3γ-val növekszik. □

Az előző tételek eredményeit foglalja össze az alábbi

15. TÉTEL. *Legyen* $k \geq 8$. *Ha* $k = 3\gamma$ *vagy* $k = 3\gamma + 1$ *alakú, akkor* $R_3\big(\mathrm{LPT}(k)\big) = \frac{5+k}{3+k}$, *és* $R_3\big(\mathrm{LPT}(k)\big) = \frac{4+k}{2+k}$ *ha* $k = 3\gamma + 2$ *alakú.* □

6. $m = 4$ gép esete

Négy gép esetén az LPT algoritmus éles felső becslése $\frac{15}{12}$. Az LPT(k) algoritmus ugyanezt az elhelyezést valósítja meg $3 \leq k \leq 8$, $k \neq 5$ esetén (a $k = 5$ esetet a 6. Tételben külön megvizsgáltuk, ekkor az éles becslés értéke $\frac{11}{9}$). Ezért a $\frac{15}{12}$ éles hatékonysági becslés adódik az 5. Tétel szerint, ha $3 \leq k \leq 4$, és ha $6 \leq k \leq 8$ (7. ábra).



7. ábra

A 7. Tétel szerint $R_4\big(\mathrm{LPT}(8 + 4\gamma)\big) = \frac{15+4\gamma}{12+4\gamma}$.

$R_4\big(\mathrm{LPT}(9 + 4\gamma)\big) = \frac{16+4\gamma}{13+4\gamma}$. A Graham-féle (2) példa esetén LPT(9) optimális megoldást ad, és eljárásunk hatékonysága $m = 4$ esetén most először jobb, mint az LPT hatékonysága. $\gamma = 0$ esetén a 7. ábra mutatja, hogy az elméleti hatékonyság legalább $\frac{16}{13}$, másrészt a 7. Tétel szerint legföljebb ennyi. Nagyobb γ számok esetén az éles példát megint úgy kapjuk, hogy hozzáveszünk 4γ számú 4 időtartamú munkát az előbbi példa elemeihez, ezek a téglalapok fognak az előzőek fölött elhelyezkedni. Ezután az általános részben még nem tisztázott esetek következnek.

16. TÉTEL. $R_4\big(\mathrm{LPT}(10)\big) = \frac{17}{14}$.

Bizonyítás. A 7. ábra jobb oldali példája szerint az elméleti hatékonyság legalább ekkora. Tegyük fel, hogy $q = 14$ egység, míg $p > 17$. Most is igaz, hogy $A > 4$.

Minden optimális gépen legföljebb 3 munka van, ezért 11 vagy 12 a téglalapok száma. A $*$ időpontban egyik gépen sem lehetett háromnál több munka. Az első tíz téglalap lerakásakor ezért 3, 3, 2, 2 vagy pedig 3, 3, 3, és 1 munka került a gépekre. Ha 12 a munkák száma, akkor minden optimális gépen három munka van, ezért minden munka ideje kisebb, mint 6. Ezért a $*$ időpontban amelyik gépen két munka van, ott az átfutási idő kisebb, mint 12, ahol viszont három, ott ennél több. Így a maradék két munka olyan gépekre kerül, ahol legfeljebb két munka van még. Ebből kapjuk, hogy $2C_1 + A > 17$, másrészt $C_1 + 2A \leq 14$, ezekből $A < \frac{11}{3}$ következik, ami ellentmondás. Tegyük fel ezért, hogy 11 a munkák száma. Ha 3, 3, 3, 1 munka került a gépekre, legyen C annak a gépnek az átfutási ideje, ahol egy munka van. Ekkor $C + A > 17$, emiatt ez a munka egyedül van az optimális megoldásál is, így a munkák száma legföljebb $3 \cdot 3 + 1 = 10$, ami túl kevés. Maradt az a lehetőség, hogy 3, 3, 2, 2 a gépeken levő munkák száma. A harmadik gépen lévő két munka legyen A_1 és A_2, az utolsó gépen levők pedig B_1 és B_2. Legyen $A_1 \geq A_2$ és $B_1 \geq B_2$. Ha az A_1 munka mellett az optimális megoldásnál még lenne két munka, akkor az előbbi egyenlőtlenség-rendszert kapnánk: $2A_1 + A > 17$, $A_1 + 2A \leq 14$, ami ellentmondásra vezet. Ezért A_1 mellett az optimális megoldásnál legfeljebb egy munka lehet, és ugyanez igaz B_1-re is. Ez csak úgy lehet, ha A_1 és B_1 ugyanazon az optimális gépen vannak és nincs több azon a gépen. A_1 és B_1 együttes ideje legfeljebb 14, ezért egyikük nem hosszabb, mint 14 fele, legyen például $A_1 \leq 7$. Ekkor $A_1 + A_2 + A > 17$, amiből kapjuk: $A_2 + A > 10$. Másrészt $A_2 + 2A \leq 14$, amiből következik $A < 4$, ellentmondás. □

Az előbbiekhez hasonlóan most LPT$(10 + 4\gamma)$ hatékonyságára szeretnénk pontos becslést kapni. Az LPT(14) algoritmus esetében a várható $\frac{21}{18}$ értéktől eltérően $\frac{20}{17}$ a pontos becslés értéke.

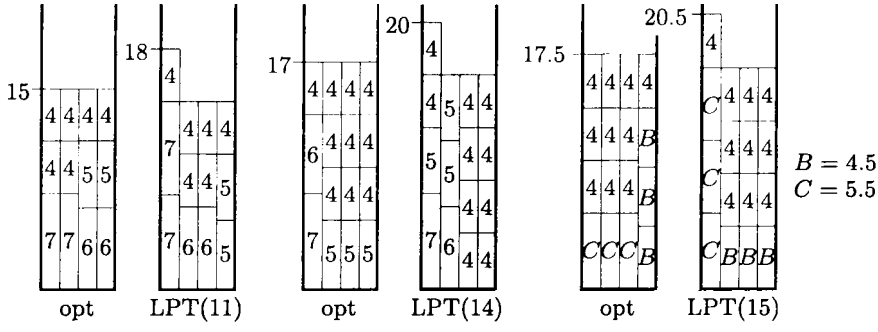
17. TÉTEL. $R_4\big(\text{LPT}(14 + 4\gamma)\big) = \frac{20+4\gamma}{17+4\gamma}$, *ahol* $\gamma \geq 1$.

Bizonyítás. Legyen $\gamma = 0$. A 8. ábrán középének példája mutatja, hogy az elméleti hatékonyság legalább ennyi. Tegyük fel, hogy $q = 17$, $p > 20$. Ekkor $A > 4$. A téglalapok száma 15 vagy 16. Könnyen látható, hogy az eljárás végén egyik gépen sem lehet 5 vagy több munka. Ha 16 téglalap van, minden optimális gépre négy munka jut, így mindegyik hossza legfeljebb 5, ezért a heurisztikus megoldás értéke legföljebb 20, ami ellentmondás. Ezért pontosan 15 munka van. A legutolsó munka olyan gépre kerül, ahol négynél kevesebb munka van. Legyen most is $A = 4 + \varepsilon$, és a 8. Tételben leírtakhoz hasonlóan ellentmondáshoz jutunk. Ezután $\gamma > 0$ esetén a szokásos módosításból: 4γ darab 4 egység hosszú munkát a feladathalmazhoz hozzávéve adódik az állítás azon része, hogy az elméleti hatékonyság értéke legalább ekkora. Az ellenkező irányú egyenlőtlenség a 10. Tételből adódik. □

Az általános részben láttuk, hogy $R_4\big(\text{LPT}(11)\big) = \frac{18}{15}$. A 8. ábrán bal oldali példája bizonyítja az élességet. Hátra van még az LPT$(11 + 4\gamma)$ algoritmus hatékonyságának megkeresése. Az LPT(15) algoritmus esetében $\frac{22}{19}$ helyett meglepő módon $\frac{20.5}{17.5}$ a pontos becslés értéke.

18. TÉTEL. *Négy gép esetén* LPT(15) *elméleti hatékonysága* $R_4\big(\mathrm{LPT}(15)\big) = \frac{20.5}{17.5}$.

Bizonyítás. A 8. ábra jobb oldali példája szerint az elméleti hatékonyság legalább ekkora. Tegyük fel, hogy $q = 17.5$, míg $p > 20.5$. Ekkor $A > 4$ és minden műveleti idő kisebb, mint 5.5. Pontosan 16 téglalap van. A $*$ időpontban mindegyik gépen van legalább három munka. Legyen C a legkisebb átfutási idő, ekkor teljesülnek az alábbi egyenlőtlenségek: $C + A > 20.5$, másrészt $\frac{C}{3} + 3A \leq 17.5$, amiből következik $A < 4$, ellentmondást kaptunk. □



8. ábra

19. TÉTEL. *Ha* $\gamma \geq 1$, *akkor* $R_4\big(\mathrm{LPT}(15 + 4\gamma)\big) = \frac{20+4\gamma}{17+4\gamma}$.

Bizonyítás. A 4 darab 5 hosszú és 16 darab 4 hosszú téglalapból álló példa ekkora hatékonyságot ad. Az optimum értéke 21, ahol minden gépen ugyanolyan munkák vannak. LPT(19) 24 egységnyi magasságot használ fel. A hatékonyság értéke legfeljebb ekkora a 10. Tétel miatt. □

A „nagy" k számokra vonatkozó becsléseket foglalja össze a

20. TÉTEL. *Ha* $k \geq 16$, *akkor* (i) $R_4\big(\mathrm{LPT}(k)\big) = \frac{7+k}{4+k}$, *ha* $k = 4\gamma$ *vagy* $k = 4\gamma + 1$ *alakú*, (ii) $R_4\big(\mathrm{LPT}(k)\big) = \frac{6+k}{3+k}$, *ha* $k = 4\gamma + 2$ *alakú*, (iii) $R_4\big(\mathrm{LPT}(k)\big) = \frac{5+k}{2+k}$, *ha* $k = 4\gamma + 3$ *alakú*. □

A következő táblázatban az $m = 2, 3, 4$ esetén kapott eredményeket szemléltetjük. Ha a $k > 23$, a táblázat alsó részén fellelhető szabály szerint következnek az elméleti hatékonyság pontos értékei.

	$m=2$	$m=3$	$m=4$
$k=1$	7/6	11/9	15/12
$k=2$	7/6	11/9	15/12
$k=3$	7/6	11/9	15/12
$k=4$	7/6	11/9	15/12
$k=5$	8/7	11/9	11/9
$k=6$	9/8	11/9	15/12
$k=7$	10/9	12/10	15/12
$k=8$	11/10	13/11	15/12
$k=9$	12/11	14/12	16/13
$k=10$	13/12	15/13	17/14
$k=11$	14/13	15/13	18/15
$k=12$	15/14	17/15	19/16
$k=13$	16/15	18/16	20/17
$k=14$	17/16	18/16	20/17
$k=15$	18/17	20/18	41/35
$k=16$	19/18	21/19	23/20
$k=17$	20/19	21/19	24/21
$k=18$	21/20	23/21	24/21
$k=19$	22/21	24/22	24/21
$k=20$	23/22	24/22	27/24
$k=21$	24/23	26/24	28/25
$k=22$	25/24	27/25	28/25
$k=23$	26/25	27/25	28/25

Az $R_m(\mathrm{LPT}(k))$ értékek

7. Az algoritmuscsalád numerikus vizsgálata

Illusztráció céljából az alábbiakban közlünk néhány futási eredményt, ahol a cikkben szereplő algoritmusokat hasonlítottuk össze. Egy-egy feladatosztályon belül vizsgáltuk az algoritmusok működését, ahol m a gépek számát, n a feladatok számát jelenti, amelyeknek az időtartamát a $[p_1, p_2]$ intervallumból választottuk egyenletes eloszlás szerint, kerekítéssel. Azt vizsgáltuk, hogy az egyes algoritmusok száz esetből hányszor adtak minimális eredményt. Az első táblázat annak illusztrálására szolgál, hogy mi történik, ha a gépek száma rögzített, és a k paramétert növeljük. A felső sorban az LPT algoritmus mellett a k paraméter növekvő értékei szerepelnek.

	m	n	p_1, p_2	LPT	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1.	2	29	9,18	0	2	0	8	8	5	8	69	75
2.	2	21	9,18	0	6	0	12	2	91	10	10	0
3.	2	16	9,18	94	71	96	26	96	91	98	65	98
4.	3	17	9,18	1	1	2	0	2	24	12	23	87
5.	3	29	9,18	1	0	0	4	1	0	1	0	100
6.	3	30	9,18	97	79	60	73	96	14	78	96	76

Mi történik a k paraméter növekedése esetén?

Az első három esetben kettő, a következő három példában három volt a gépek száma. A munkák időtartamát mindegyik esetben a $[9, 18]$ intervallumból választottuk. Az első és negyedik sorban arra látunk példát, amikor az algoritmus hatékonysága a k paraméter növekedésével együtt növekszik. A 2. és 5. példa azt illusztrálja, hogy az előbbi szituáció nem általános: az LPT(k) hatékonyságát erősen befolyásolja a munkák számának és a k paraméternek az oszthatósági viszonya. Amikor k osztója a feladatok számának, LPT(k) hatékonyan meg tudja oldani a feladatot (mint például a 2. sorban $k = 7$ esetén), vagy például akkor is, ha a maradék -1 (5. sor, $k = 10$ esete). Ilyen irányú alaposabb vizsgálatokat (ami a k paraméter és a munkák n számának oszthatósági viszonyait érinti) a jelen cikk keretei között nem végeztünk. A 3. és 6. példa különleges abban, hogy itt minden második, illetve a 6. esetében minden harmadik k-ra majdnem mindig optimális megoldást kapunk, a többi k-ra pedig ezeknél rosszabb megoldásokat.

	m	n	p_1, p_2	LPT	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1.	2	17	9,18	0	0	0	0	0	0	0	100	10
2.	3	20	9,18	0	0	0	0	0	3	3	0	98
3.	4	18	9,18	0	1	0	0	4	9	0	97	28

Egy speciális eset: k értéke a munkák számának a fele.

A második táblázatunk egy érdekes jelenséget mutat. A k paramétert úgy választottuk meg, hogy akkora legyen, mint a munkák számának a fele. Az esetek túlnyomó részében ekkor az algoritmusunk optimális megoldást adott.

Az algoritmus bonyolultsága a k paraméter növelésével exponenciálisan növekszik. A számítógépes realizáció során az algoritmus általános lépésénél, tehát a következő k számú munka helyének a keresésénél korlátozás és szétválasztás típusú algoritmust alkalmaztunk. Ennek során, a következő k számú munka optimális elhelyezése után adódó teljes átfutási időre egy egyszerű felső becslést használtunk: a következő k darab munkának az LPT algoritmus szerinti elhelyezésével adódó

becslést. A program futása például öt gép, $k = 10$ és $n = 30$ munka esetén Pentium 1-es típusú géppel körülbelül négy másodpercig tartott, négy gép esetén pedig körülbelül fél másodpercig. Ez azt mutatja, hogy kis vagy közepes méretű feladat esetében az algoritmus számítógépes futása hamar véget ér. A kapott eredményeket összefoglalva azt mondhatjuk, hogy a k paraméter növelésével az algoritmus hatékonysága bizonyos esetekben jelentősen növekszik, de jelentősen befolyásolja a hatékonyságot a k és n oszthatósági viszonya.

Hivatkozások

[1] Dósa, Gy. and Vizvári, B., Az Általánosított LPT $(k)'$ algoritmus egyforma párhuzamos gépek ütemezésére, *Alkalmazott Matematikai Lapok* **21** (2004) 269–290.

[2] Dósa, Gy., Graham's example is the only tight one for $P\|C_{\max}$, *Annales Univ. Sci. Budapest.* **47** (2004) 143–146.

[3] Graham, R. L., Bounds for certain multiprocessor anomalies, *Bell System Tech. J.* **45** (1966) 1563–1581.

[4] Graham, R. L., Bounds on multiprocessor timing anomalies, *SIAM J. Appl. Math.* **17** (1969) 416–429.

[5] Vizvári, B., Bevezetés a termelésirányítás matematikai elméletébe, *Egyetemi jegyzet* (ELTE, Budapest, 1992).

(Beérkezett: 2001. március 26.)

DÓSA GYÖRGY
VESZPRÉMI EGYETEM
dosagy@almos.vein.hu

VIZVÁRI BÉLA
ELTE OPERÁCIÓKUTATÁSI TANSZÉK
vizvari@cs.elte.hu

THE GENERAL ALGORITHM LPT(k) FOR SCHEDULING IDENTICAL PARALLEL MACHINES

GYÖRGY DÓSA AND BÉLA VIZVÁRI

The paper is devoted to investigate the general algorithm LPT(k). First the tasks are ordered to the LPT order. Then at the same time commonly k tasks are scheduled in a (locally) optimal way (when the increase of the makespan is minimal), and this step is iterated. As our main result, we give the tight value of the theoretical efficiency of the algorithm for every k and every $2 \leq m \leq 4$, where m is the number of machines. Numerical results are also given.