Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Számítástudományi és Információelméleti Tanszék

Magyar Tudományos Akadémia
Számítástechnikai és Automatizálási Kutatóintézet
Mérnöki és Üzleti Intelligencia Kutatólaboratórium

Gépütemezés erőforráskorlátokkal

Drótos Márton

PhD tézisfüzet

Témavezető:

Wiener Gábor

Számítástudományi és Információelméleti Tanszék
Villamosmérnöki és Informatikai Kar
Budapest Műszaki és Gazdaságtudományi Egyetem
Budapest, Magyarország

Konzulens:

Kis Tamás

Mérnöki és Üzleti Intelligencia Kutatólaboratórium
Számítástechnikai és Automatizálási Kutatóintézet
Magyar Tudományos Akadémia
Budapest, Magyarország

Gépütemezés erőforráskorlátokkal

PhD tézisfüzet

Bevezetés

A gépütemezés egy, a gyakorlatban is jól alkalmazható, népszerű témája az operációkutatásnak. Az utóbbi évtizedekben számos modell és algoritmus készült a modern iparban felmerülő ütemezési problémák megoldására. Bár a legtöbb ilyen probléma NP-nehéz, az elérhető számítási teljesítmény növekedése lehetővé teszi, hogy egyre több és több részlettel egészítsük ki a modelljeinket. A gazdasági környezet is változik: a termelés egyre inkább eltolódik tömegtermelés irányából az egyedi, rendelésre gyártás felé. Az ipari igények változását az ütemezési modelleknek és algoritmusoknak is követniük kell: a fő optimalizálási szempontok megváltoznak (pl. az ügyfélkiszolgálás magas színvonalon tartása fontosabbá válik az átfutási idők minimalizálásánál), valamint a szűk keresztmetszetek is mások lesznek (pl. drágább a munkaerő, a raktározás, stb.).

Munkám során ezeket az igényeket igyekeztem szemem előtt tartani, vagyis olyan gépütemezési problémákat vizsgáltam, amikben figyelembe kell venni megújuló (pl. operátorok, munkaeszközök), valamint nem megújuló (pl. alapanyagok) erőforrásokat. Emellett kutatásom során az is szempont volt, hogy közvetlen ipari felhasználásra alkalmas megoldásokat adjak. A kutatási eredményeimet 3 tézisben fogalmaztam meg.

1. Tézis. *Egygépes környezetben vizsgáltam késés jellegű célfüggvények minimalizálását készletkitárolást végző munkák esetén. Ebben a modellben a munkák különféle erőforrásokat termelnek, és a határidővel rendelkező kiszállítások ezekre az erőforrásokra tartanak igényt. A problémára bonyolultságelméleti eredményeket, valamint pszeudopolinomiális és approximációs algoritmusokat adtam.*

Bebizonyítottam, hogy amennyiben minden munka azonos mennyiségű erőforrást termel, vagy csak egy típusú erőforrás van és minden munkának azonos a végrehajtási ideje, akkor a probléma polinomidőben megoldható (1.1. tétel és 1.2. tétel). Azt is bizonyítottam továbbá, hogy a probléma erősen NP-nehéz még akkor is, ha minden munka végrehajtási ideje azonos és legalább kétféle erőforrástípus vagy két kiszállítás van (1.3. tétel és 1.4. tétel), valamint hagymányos értelemben NP-nehéz, amennyiben legalább egy erőforrástípus és legalább két kiszállítás van (1.5. tétel).

Ezentúl adtam egy pszeudopolinomiális algoritmust arra a speciális esetre, amikor az erőforrástípusok és a kiszállítások száma konstans (1.6. tétel), valamint egy teljesen polinomidejű approximációs sémát arra a speciális esetre, amikor egyféle erőforrás és két kiszállítás van (1.7. tétel).

2. Tézis. *Az erőforráskiegyenlítési problémára adtam korlátozás és elágazás algoritmust dedikált párhuzamos gépek esetén (2.1. algoritmus). Az algoritmus hatékonyságát számítási kísérletekkel támasztom alá úgy, hogy a véletlenszerűen generált problémákra adott eredményeit összevetem egy általános, kereskedelmi forgalomban kapható megoldó eredményeivel.*

A problémát felírtam egészértékű programozási problémaként, és ezt a felírást felhasználva egy Lagrange-relaxációt használó algoritmust adtam alsó korlátok számítására (2.1. tétel).

A kapott alsó korlátok javítására és a keresési fa méretének csökkentésére egy, a Lagrange-relaxáción alapuló „shaving" eljárást dolgoztam ki (2.2. tétel és 2.3. tétel).

Adtam továbbbá egy módszert felső korlát számítására is a keresési fa csúcsaiban (2.1. lemma és 2.2. algoritmus).

Egy újfajta elágazási sémát is kidolgoztam, ami a korlátozás és elágazás algoritmus hatékonyságát növeli (2.4.5. szakasz).

3. Tézis. *Leírtam egy komplex ipari ütemezési problémák megoldására alkalmas keretrendszert (3.1. algoritmus), és különféle algoritmusokat adtam, melyek a keretrendszer egyes fázisaiban használhatók különböző ütemezési modellek esetén, valamint újfajta szomszédságokat is készítettem lokális keresésekhez. A bemutatott keretrendszer hatékonyságát két ipari esettanulmány bemutatásával támasztom alá.*

A keretrendszer egyik fő célkitűzése a megújuló és nem megújuló erőforrások hatékony kezelése. A nem megújuló erőforrások munkákhoz rendelésére egy dinamikus programozást használó algoritmust adtam (3.2. algoritmus).

Mivel a keretrendszer nagyrészt lokális keresést végző algoritmusokat használ, két módszert is adtam az egyes munkák kezdési és befejezési időpontjának kiszámítására/újraszámítására különféle ütemezési problémák esetén. A 3.3. algoritmus a kezdési és befejezési időpontok gyors újraszámítását végzi, míg a 3.4. algoritmus képes komplex modellekben a megújuló erőforrások kiosztását is elvégezni.

Egy olyan algoritmust is készítettem, ami egyes egyszerűbb modellekben alkalmas a megújuló erőforrások kezelésére (3.5. algoritmus).

A három tézis részletes eredményeit az 1., 2. és 3. szakaszban mutatom be részletesen.

1. Készletek kitárolását végző munkák ütemezése késés jellegű célfüggvények esetén

1.1. Motiváció

Ebben a tézisben egy olyan ütemezési problémával foglalkozom, ahol a munkák érkező készletek kitárolásának felelnek meg (pl. kamionok lerakodása). Ezeket a munkákat kell egy gépen (kiszolgálón) sorrendezni úgy, hogy a készletekre vonatkozó igényeket minél jobban kielégítsük. A gépütemezésben megszokottakkal ellentétben ebben a modellben nem az egyes munkákra adottak határidők, hanem megadott időpontokban megadott készletcélokat kell teljesíteni.

Egy lehetséges felhasználási terület például egy Just-In-Time rendszerben működő gyár, ahol a termeléshez szükséges alapanyagokat közvetlenül a felhasználás előtt szállítják a beszállítók. Ha anyaghiány keletkezne, akkor a termelést vagy késleltetni kéne, vagy át kéne ütemezni. Emiatt a beérkező anyagok kirakodására olyan ütemtervet kell adni, ami figyelembe veszi a kirakodási idők, kapacitások, stb. által jelentett korlátokat. Ily módon el lehet érni, hogy a termelést a lehető legkisebb mértékben késleltethessék az esetleges anyaghiányok.

1.2. A probléma leírása

Adott a munkák egy J halmaza, valamint terméktípusok egy S halmaza, melyeknek számossága $|J| = n^J$ és $|S| = n^S$. Minden $j \in J$ munkának adott a $p_j > 0$ feldolgozási ideje, valamint minden $s \in S$ termék esetén a $\tilde{c}_j^s \geq 0$ termelése. Az is lehetséges, hogy ugyanaz a munka többféle terméket is termel. Kezdetben minden termék készlete 0. A készletet egy R halmaz által definiált kiszállítások fogyasztják, melynek mérete $|R| = n^R$. Minden $R_r \in R$ kiszállításnak adott a d_r határideje, továbbá minden $s \in S$ termékre a $\tilde{e}_r^s$ készletigénye. A munkákhoz hasonlóan a kiszállítások is igényelhetnek egyszerre többféle terméket is. Feltételezzük, hogy a kiszállítások határidő szerint rendezettek, azaz $d_1 \leq d_2 \leq \cdots \leq d_{n^R}$. Egy kiszállítás akkor végezhető el, ha minden igényelt termékből a készletszint legalább annyi, mint az igénye. Ebből következik, hogy készlethiány esetén az egyes kiszállítások késhetnek. A kiszállításokat határidő szerinti nem-csökkenő sorrendben kell teljesíteni, határidőegyezés esetén pedig előre rögzítettnek tekintjük a sorrendjüket. A kiszállítások indexelése eszerint a sorrend szerint történik.

Legyen π az n^J munka egy sorrendje, azaz $\pi(u) \in J$ meghatározza, melyik munka sze-

repel az u ($1 \leq u \leq n^J$) pozícióban, valamint $\pi(u) \neq \pi(v)$, ha $u \neq v$. Az s termék teljes beérkezett készletszintjét a π első h munkájának elvégzése után $\sum_{u=1}^{h} \tilde{c}^s_{\pi(u)}$ módon számíthatjuk. Az első r kiszállítás összigényét hasonlóképp, $e^s_r = \sum_{k=1}^{r} \tilde{e}^s_k$ módon írhatjuk fel. Jelöljük h^π_r-rel azt a minimális darabszámot, ahány munka szükséges az első r kiszállítás összigényének kielégítéséhez, azaz $h^\pi_r := \min\left\{h \mid \sum_{u=1}^{h} \tilde{c}^s_{\pi(u)} \geq e^s_r, \text{ for all } s \in S\right\}$. Ennek segítségével az r-edik kiszállítás időpontja a következőképp számítható (feltételezvén, hogy a munkák szünet nélkül követik egymást): $C^\pi_r := \sum_{u=1}^{h^\pi_r} p_{\pi(u)}$. Amennyiben $C^\pi_r \leq d_r$, az r kiszállítás pontos, egyébként késik. A késés lehetséges definíciói: $L^\pi_r = C^\pi_r - d_r$ (lateness) és $T^\pi_r := \max(0, L^\pi_r)$ (tardiness). Egy π sorrendben $\pi(h)$ munka befejezési idejét $\tilde{C}^\pi_h = \sum_{k=1}^{h} p_{\pi(k)}$ módon számíthatjuk (feltételezvén, hogy a munkák szünet nélkül követik egymást). Az s termék összkészletét a t időpontig π ütemtervben $A^s_\pi(t) = \sum_{h|\tilde{C}^\pi_h \leq t} \tilde{c}^s_{\pi(h)}$ módon számíthatjuk. A kiszállítások befejezési idejében reguláris (monoton) célfüggvények minimalizálását vizsgálom. Ilyenek például: $\max_r T_r$ (maximal tardiness), $\max_r L_r$ (maximal lateness), $\sum_r w_r T_r$ (total weighted tardiness) vagy $\sum_r w_r L_r$ (total weighted lateness). Jelöljük $\gamma(\pi)$-vel a π ütemterv célfüggvényét. Mivel mind π, mind a kiszállítások száma véges, γ jól definiált.

Az általánosság megsértése nélkül feltételezhetjük, hogy az utolsó kiszállítás határideje kielégíti a $d_{n^R} \geq \sum_j p_j$ összefüggést, továbbá hogy minden $s \in S$ termékre teljesül, hogy $e^s_{n^R} = \sum_j \tilde{c}^s_j$. Ezt azért tehetjük, mert ha $d_{n^R} < \sum_j p_j$, akkor mindig hozzávehetünk egy új kiszállítást R-hez, melynek határidejét $\sum_j p_j$-re állíthatjuk. Másrészt ekkorra már minden termék beérkezik, így feltételezhetjük, hogy $e^s_{n^R} = \sum_{j \in J} \tilde{c}^s_j$ minden $s \in S$-re.

A problémát az 1. ábrán illusztrálom.

1.3. Kapcsolódó eredmények

Ez a tézis Boysen et al. (2013) eredményeit egészíti ki. Abban a modellben a kiszállítások határidejei nem sérülhetnek, és a hivatkozott cikk olyan speciális eseteket vizsgál, ahol egy megengedett ütemterv létezése polinomidőben eldönthető, a célfüggvény pedig a készletszinteknek valamely függvénye. Az általam vizsgált probléma egy optimális vagy közel optimális megoldása felhasználható például arra, hogy ennek segítségével Boysen et al. modelljéhez határidőket szolgáltassunk, és a készletszinteket az ő módszereikkel minimalizáljuk.

Grigoriev et al. (2005) a probléma megfordítottját vizsgálta, vagyis egy olyan modellt, ahol a munkák alapanyagokat fogyasztanak a készletből. Abban a modellben a beszállítások adottak az idő függvényében, és minden munka előre meghatározott mennyiségű alapanya-

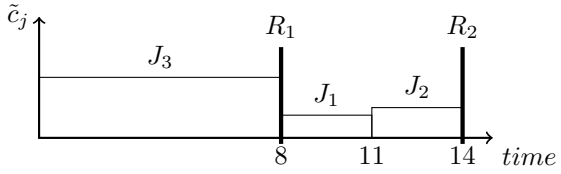
Paraméterek:

Param.	Érték
n^J	3
n^S	1
n^R	2

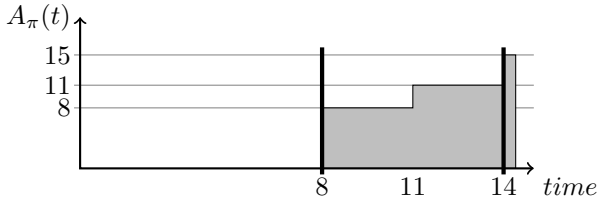
Munka	p_j	$\tilde{c}_j$
J_1	3	3
J_2	3	4
J_3	8	8

Kiszállítás	d_r	$\tilde{e}_r$	e_r
R_1	7	6	6
R_2	14	9	15

A $\pi(J_3, J_1, J_2)$ ütemterv:



$\pi(J_3, J_1, J_2)$ össztermelése:



1. ábra. Példa az ütemezési problémára. A dobozok az ütemtervben a munkákat jelképezik; a j munka hossza p_j-vel, míg magassága $\tilde{c}_j$-vel arányos. A bemutatott ütemterv maximális késése 1 időegység.

got igényel. Egy munka csak akkor végezhető el, ha elkezdésekor rendelkezésre áll minden szükséges alapanyag, és elkezdésekor a készleteket a megfelelő mennyiséggel csökkenti. A hivatkozott cikk ennek a problémának a különböző változatait vizsgálja, és NP-nehézségi eredményeket, a nehéz változatokra approximációs algoritmusokat, valamint a kezelhető változatokra polinomiális algoritmusokat mutat.

A bemutatott probléma általánosítható egy olyan modell segítségével, ahol a munkák egy közös anyaghalmazból fogyasztanak és termelnek is, és az alapkérdés az, hogy a termelőknek és fogyasztóknak létezik-e egy megengedett sorrendje. Kellerer et al. (1998) megmutatták, hogy ez a probléma erősen NP-nehéz. Briskorn et al. (2010) továbbvitték ezt a munkát, és tanulmányozták a probléma különböző változatait, melyekre NP-nehézségi bizonyításokat vagy polinomidejű algoritmusokat adtak.

1.4. Eredmények

Elsőként bonyolultságelméleti eredményeket adtam az $1||\gamma$ problémára, továbbá polinomiális és pszeudopolinomiális algoritmusokat készítettem minden olyan esetre, amire ez

		$n^S = 1$			$n^S \geq 2$, konst.			$n^S = *$	
		$\tilde{c}_j^s = \tilde{c}^s$	$\tilde{c}_j^s = *$		$\tilde{c}_j^s = \tilde{c}^s$	$\tilde{c}_j^s = *$		$\tilde{c}_j^s = \tilde{c}^s$	$\tilde{c}_j^s = *$
	$p_j = 1$	P (1.2)	P	(1.1)	P (1.2)	o.NP[a]		P (1.2)	s.NP (1.3)
						p.poli.	(1.6)		
$n^R \geq 2$, konst.	$p_j = p$	P (1.2)	P	(1.1)	P (1.2)	o.NP[a]		P (1.2)	s.NP (1.3)
						p.poli.	(1.6)		
	$p_j = *$	P (1.2)	o.NP	(1.5)	P (1.2)	o.NP[a]		P (1.2)	s.NP (1.3)
			p.poli.	(1.6)		p.poli.	(1.6)		
	$p_j = 1$	P (1.2)	P	(1.1)	P (1.2)	s.NP	(1.4)	P (1.2)	s.NP (1.3)
$n^R = *$	$p_j = p$	P (1.2)	P	(1.1)	P (1.2)	s.NP	(1.4)	P (1.2)	s.NP (1.3)
	$p_j = *$	P (1.2)	s.NP[b]		P (1.2)	s.NP	(1.4)	P (1.2)	s.NP (1.3)

A táblázatban a következő, másoktól származó eredményeket is feltüntettem:
[a] $1|p_j = 1, n^S = 2, n^R = 2|\gamma^T$ NP-nehézségének bizonyítása a (Drótos és Kis; 2013a) cikkben található
[b] $1|n^S = 1|\gamma^T$ erős NP-nehézségének bizonyítása a (Boysen et al.; 2013) cikkben található

1. táblázat. Az $1||\{\gamma, \gamma^T\}$ bonyolultságelméleti eredményeinek áttekintése. Minden mező a probléma egy speciális esetének felel meg; a csillag ($*$) azt jelenti, hogy az adott paraméter bármilyen pozitív egész szám lehet. A bonyolultság vagy polinomiális (P), vagy NP-nehéz a hagyományos értelemben (o.NP), vagy erősen NP-nehéz (s.NP). Pszeudopolinomiális algoritmus létezését p.poli. jelöli. A γ célfüggvény a kiszállítások befejezési idejének bármely reguláris függvénye a polinomiális esetekben, a γ^T célfüggvény pedig a kiszállítások késésének bármely reguláris függvénye az NP-nehéz esetekben. A zárójelben szereplő számok az eredményt bizonyító tétel számát jelentik.

lehetséges (feltéve, hogy $P \neq NP$). Ezentúl egy teljesen polinomidejű approximációs sémát (FPTAS) is kidolgoztam az egyik esetre. Az eredmények azt mutatják, hogy már az általános probléma speciális eseteinek nagy részét is kifejezetten nehéz megoldani. A 1. táblázat foglalja össze a bonyolultságelméleti eredményeket.

1.1. Tétel. *Az* $1|p_j = p, n^S = 1|\gamma$ *probléma* $O(n^J \log n^J)$ *időben megoldható.*

1.2. Tétel. *Az* $1|\tilde{c}_j^s = \tilde{c}^s|\gamma$ $O(n^J \log n^J)$ *időben megoldható.*

1.3. Tétel. *Az* $1|n^R = 2, p_j = 1|\gamma^T$ *erősen NP-nehéz.*

1.4. Tétel. *Az* $1|n^S = 2, p_j = 1|\gamma^T$ *erősen NP-nehéz.*

1.5. Tétel. *Az* $1|n^S = 1, n^R = 2|\gamma^T$ *a hagyományos értelemben NP-nehéz.*

1.6. Tétel. *Az* $1|n^S = const., n^R = const.|\gamma$ *probléma megoldható pszeudopolinomiális időben.*

1.7. Tétel. *Az* $1|n^S = 1,\ n^R = 2|T_{\max}$ *problémára készíthető FPTAS.*

1.5. Publikációk

Az 1. tézis eredményeit a (Drótos és Kis; 2013a,b; Drótos et al.; 2013) publikációkban tettem közzé.

2. Erőforráskiegyenlítés párhuzamos gépek esetén

2.1. Motiváció

Megújuló kumulatív erőforrásokkal nagyon jól lehet modellezni egy gyár munkaerőigényét. Tekintsünk egy olyan gyárat, ahol vannak állandó és átmeneti munkások. Amennyiben a gyár termelése ingadozik az idő folyamán, elképzelhető, hogy a csúcsidőszakokban nincs elég állandó alkalmazott az igények kielégítéséhez. Ebben az esetben kívülről kell bérelni munkaerőt, és a bérlés költségének minimalizálására egy erőforráskiegyenlítési problémát kell megoldani megfelelő célfüggvény használatával.

Egy tipikus ütemezési probléma megoldása során, amikor a munkaerőigényekkel foglalkozunk, a megoldás egyes paraméterei már sokszor rögzítettnek tekinthetők (pl. a határidők betartásának kényszere miatt). Ez azt eredményezi, hogy az egyes munkák gépekhez rendelése már megtörtént, valamint minden munkára adott egy időablak, amiben azt el kell ahhoz végezni, hogy ne sértsen meg egyéb korlátozásokat. Ebben a tézisben erre a problémára adok egy egzakt módszert, amivel optimális, vagy közel optimális ütemtervek készíthetők.

2.2. A probléma leírása

Adott m darab párhuzamos gép, $M_1, \ldots, M_m$, L fajta megújuló erőforrás, $R_1, \ldots, R_L$, valamint minden erőforráshoz egy célszám, $C_1, \ldots, C_L \in \mathbb{Q}$. Adott továbbá n munka, melyek egy J halmazt alkotnak. Ez a halmaz m diszjunkt részre van particionálva: $J = J_1 \cup J_2 \cdots \cup J_m$. A J_i partícióban lévő összes munkát az M_i gépen kell elvégezni. Minden j munkához adott továbbá egy pozitív egész megmunkálási idő (p_j), legkorábbi kezdési idő (e_j), egy határidő (d_j), valamint minden erőforrástípusból az igényelt mennyiség ($b_{j1}, \ldots, b_{jL}$). Az egyes munkák megszakítása nem megengedett. Semelyik gép nem végezhet egyszerre több munkát, viszont az erőforrások párhuzamos használatára nincsen korlátozás. Egy $\mathcal{S}$ ütemterv meghatározza minden munka S_j kezdési idejét, vagyis $\mathcal{S} = (S_1, \ldots, S_n)$. Egy ütemterv akkor megengedett, ha minden j munkára teljesülnek az $e_j \leq S_j$, valamint $S_j \leq d_j - p_j$ feltételek, valamint az azonos gépen elvégzett semelyik két j és k munka feldolgozása sem lapolódik át ($[S_j, S_j + p_j) \cap [S_k, S_k + p_k) = \emptyset$ bármely $j, k \in J_i, j \neq k$ esetén).

A célfüggvények felírásához további definíciókra van szükség. Jelölje D azt a legkésőbbi időpontot, ahol még valamely munka elkezdődhet, azaz $D = (\max_{j \in J} d_j) - 1$. Egy $\mathcal{S}$ ütem-

terv erőforrásprofilja egy olyan $R^{\mathcal{S}} = (r_1^{\mathcal{S}}, \ldots, r_L^{\mathcal{S}})$ függvény, ahol $r_\ell^{\mathcal{S}} : [0, D+1] \to \mathbb{Q}_+$ a t időpontbeli összigényt adja meg az ℓ erőforrásból $\mathcal{S}$ ütemtervben, vagyis $r_\ell^{\mathcal{S}}(t) = \sum_{j \in J: S_j \le t < S_j + p_j} b_{j\ell}$. Jelen tézisben a következő formában adott célfüggvényekkel foglalkozom:

$$f(R^{\mathcal{S}}) := \sum_{\ell=1}^{L} \sum_{t=0}^{D} \hat{f}_\ell(r_\ell^{\mathcal{S}}(t), C_\ell),$$

ahol minden $\hat{f}_\ell : \mathbb{Q}_+ \times \mathbb{Q}_+ \to \mathbb{Q}_+$ függvény kielégíti az $\hat{f}_\ell(x, y-z) = \hat{f}_\ell(x+z, y)$ összefüggést. Példának okáért, ha $\hat{f}_\ell(x, y) = w_\ell \max\{0, x-y\}$ $(w_\ell \ge 0)$ függvényt választunk, a következő célfüggvényt kapjuk:

$$f_{\text{lin}}(R^{\mathcal{S}}) = \sum_{\ell=1}^{L} w_\ell \sum_{t=0}^{D} \max\{0, r_\ell^{\mathcal{S}}(t) - C_\ell\},$$

Amennyiben $\hat{f}_\ell(x, y) = w_\ell (x-y)^2$, akkor

$$f_{\text{quad}}(R^{\mathcal{S}}) = \sum_{\ell=1}^{L} w_\ell \sum_{t=0}^{D} (r_\ell^{\mathcal{S}}(t) - C_\ell)^2.$$

Természetesen az egyes ℓ erőforrástípusokra nem szükséges, hogy az $\hat{f}_\ell$ függvények megegyezzenek.

2.3. Kapcsolódó eredmények

Hagyományosan az erőforráskiegyenlítési problémával a projektütemezés foglalkozik. Az egyik első heurisztikus megoldást erre a problémára Burgess és Killebrew (1962) készítette. A hivatkozott megoldás CPM/PERT hálózatokra alkalmazható, amelyekben a csúcsok felelnek meg a munkáknak, míg az élek az egyes munkák közötti időbeli kényszereket jelképezik. Az erőforráskorlátos projektütemezés frissebb eredményeinek összefoglalása megtalálható a (Hartmann és Briskorn; 2010) cikkben, a (Demeulemeester és Herroelen; 2002) könyv pedig további modellek és algoritmusok széles választékát mutatja be.

Az erőforráskorlátos gépütemezés témakörének is van nem elhanyagolható irodalma. Az ilyen problémák osztályozásának módszertanát Błażewicz et al. (1983) alkották meg, és emellett ezekre a problémákra bonyolultságelméleti vizsgálatokat is végeztek. Az ilyen típusú ütemezési problémákról és a téma további irodalmáról lásd a (Błażewicz; 2001) könyvet.

Tudomásom szerint jelenleg nincsen olyan eredmény, ami gépütemezési problémákban foglalkozna erőforráskiegyenlítéssel.

Mivel a bemutatott modellben már egy gépre is NP-nehéz annak eldöntése, hogy létezike megengedett ütemterv ([SS1] probléma, Garey és Johnson; 1979), a probléma bonyolultságelméleti analízise nem mutatná jól, milyen nehézséget jelent az erőforrások kezelése. Ennek elemzése megtalálható a (Drótos és Kis; 2011) cikkben.

2.4. Eredmények

A bemutatott probléma megoldására egy olyan korlátozás és elágazás (branch-and-bound) algoritmust dolgoztam ki, amelyben az egyes csúcsok az eredeti feladat egy megszorított változatát jelképezik. Ez a megszorítás azt jelenti, hogy az egyes csúcsokban valamely munkák időablakát szűkítem, a gyökér pedig az eredeti időablakokat tartalmazza. A keresési fa csúcsainak sematikus bemutatása a 2. ábrán látható.

2.1. Algoritmus. *A keresési fa egy csúcsának kiértékelése a következőképp történik:*

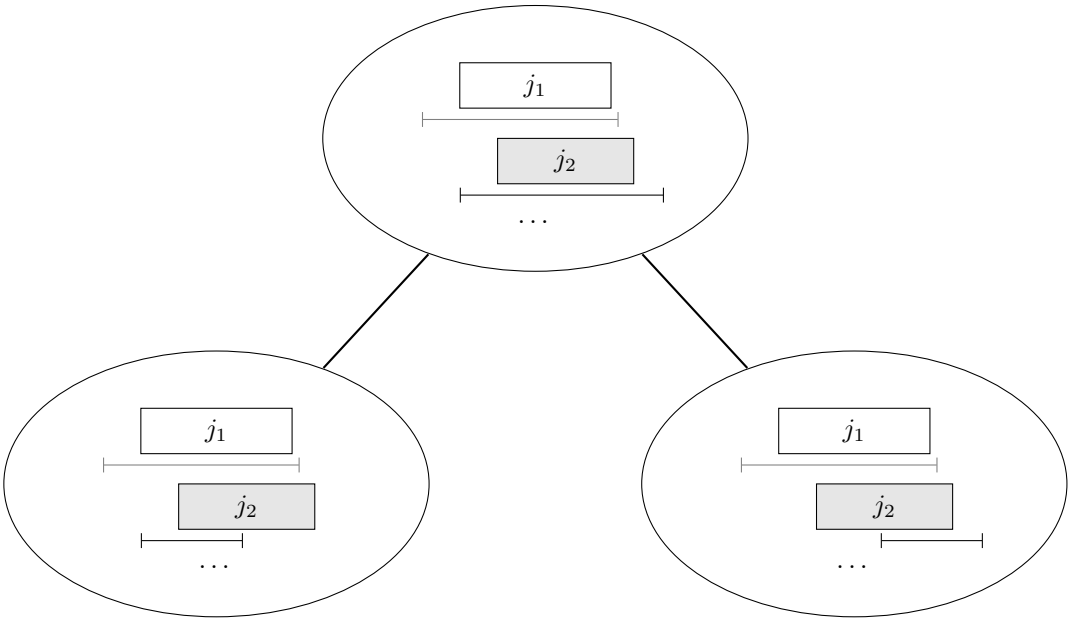
1. *Korlátpropagálás*
2. *Alsó korlát számítása*
3. *Shaving*
4. *Felső korlát számítása*
5. *Elágazás*

2.4.1. Korlátpropagálás

Az időablakok szűkítésére az egygépes ütemezés irodalmából jól ismert korlátpropagálási eljárásokat használok gépenként. Amennyiben kiderül, hogy valamely gépen nem létezhet megengedett megoldás, akkor a keresési fa adott csúcsa elvethető, mert az abban definiált problémának nem létezhet megengedett megoldása.

2.4.2. Alsó korlát számítása

A célfüggvény alsó korlátjának számítása a feladat egészértékű programozási problémaként (IP) való felírásán alapul. A következő döntési változókat használom: $x_{jt} \in \{0, 1\}$,



2. ábra. A korlátozás és elágazás fájának illusztrációja. Minden elágazási lépés az egyik munka időablakát kisebb intervallumokra particionálja. A példa azt az esetet mutatja be, amikor a j_2 munka időablakát két kisebb intervallumra osztja az algoritmus, így hozva létre két, a szülő csúcshoz képest több megszorítást tartalmazó problémát.

$j \in J$, $t \in \{e_j, \ldots, d_j - p_j\}$, valamint $y_{\ell t} \in \mathbb{R}$, $j \in J$, $t \in \{0, \ldots, D\}$. Az x_{jt} változók az egyes munkák kezdési időpontját adják meg az $\mathcal{S}$ ütemtervben, míg az $y_{\ell t}$ változók az $r_\ell^{\mathcal{S}}(t)$ erőforráshasználat kiszámítását végzik.

$$\min \sum_{\ell=1}^{L} \sum_{t=0}^{D} \hat{f}_\ell(y_{\ell t}, C_\ell) \tag{1}$$

feltéve, hogy

$$\sum_{t=0}^{D} x_{jt} = 1, \quad \forall\, j \in J, \tag{2}$$

$$\sum_{j \in J_i} \sum_{\tau=t-p_j+1}^{t} x_{j\tau} \leq 1, \quad \forall\, t \in \{0, \ldots, D\},\ i \in \{1, \ldots, m\} \tag{3}$$

$$\sum_{j \in J} \sum_{\tau=t-p_j+1}^{t} b_j x_{j\tau} - y_{\ell t} = 0, \quad \forall\, t \in \{0, \ldots, D\},\ \ell \in \{1, \ldots, L\} \tag{4}$$

$$x_{jt} \in \{0, 1\}, \quad \forall j \in J,\ t \in \{e_j, \ldots, d_j - p_j\}. \tag{5}$$

A (1)-(5) Lagrange-relaxációját a (4) korlátok dualizálásával készítem el. Mivel mind a célfüggvény, mind a megfelelő korlátok szétválaszthatók, a relaxáció a következő matematikai programot eredményezi:

$$LB(\boldsymbol{\lambda}) = \sum_{i=1}^{m} LB_i(\boldsymbol{\lambda}) + \min_{y} \sum_{\ell=1}^{L} \sum_{t=0}^{D} \left(\hat{f}_\ell(y_{\ell t}, C_\ell) - \lambda_{\ell t} y_{\ell t} \right), \tag{6}$$

ahol

$$LB_i(\boldsymbol{\lambda}) = \min \sum_{\ell=1}^{L} \sum_{t=0}^{D} \sum_{j \in J_i} \sum_{\tau=t-p_j+1}^{t} \lambda_{\ell t} b_{j\ell} x_{j\tau} \tag{7}$$

feltéve, hogy

$$\sum_{t \in \{e_j, \ldots, d_j - p_j\}} x_{jt} = 1, \quad \forall\, j \in J_i, \tag{8}$$

$$\sum_{j \in J_i} \sum_{\tau=t-p_j+1}^{t} x_{j\tau} \leq 1, \quad \forall\, t \in \{0, \ldots, D\} \tag{9}$$

$$x_{jt} \in \{0,1\}, \quad \forall j \in J_i,\ t \in \{e_j, \ldots, d_j - p_j\}. \tag{10}$$

Az $LB_i(\boldsymbol{\lambda})$ részproblémák továbbra is nehezen megoldható egészértékű programozási feladatok, ezért ezeknek a lineáris relaxációját (LP) oldom meg. Ez annyit tesz, hogy az $x_{jt} \in \{0,1\}$ korlátokat $0 \leq x_{jt} \leq 1$ korlátokra cserélem.

2.1. Tétel. *$LB(\boldsymbol{\lambda})$ alsó korlátot ad az erőforráskiegyenlítési feladat célfüggvényére bármely $\boldsymbol{\lambda} = (\lambda_{\ell t} : \ell \in \{1, \ldots, L\}, t \in \{0, \ldots, D\})$ felhasználásával, továbbá ez a korlát rögzített $\boldsymbol{\lambda}$ esetén m darab független egygépes ütemezési probléma megoldásával számítható ki.*

A bemutatott megközelítés nagy előnye, hogy az (7)-(10) által meghatározott LP-k mérete független az erőforrástípusok darabszámától és a célfüggvénytől is. Mivel az LP-k mérete $O(Dn_i)$-vel becsülhető, megoldásuk a probléma méretében pszeudopolinomiális időben elvégezhető.

Az $LB = \max_{\boldsymbol{\lambda}} LB(\boldsymbol{\lambda})$ optimális alsó korláthoz tartozó Lagrange-szorzók approximálására a szubgradiens-módszert használom, az adott célfüggvény által esetlegesen támasztott kényszerek figyelembevételével.

2.4.3. Shaving

A shaving eljárás (Carlier és Pinson; 1994; Martin és Shmoys; 1996) egy hatékony korlátpropagálási technika, ami a döntési változók megengedett tartományának szűkítésére való választási pontok létrehozása nélkül. A módszer lényege, hogy egy döntési változót rögzítünk valamely megengedett értékére, majd a megfelelő korlátpropagálási lépések elvégzése után, amennyiben nem megengedett megoldást kaptunk, kizárjuk ezt az értéket. A shaving technika részleteiről lásd (Torres és Lopez; 2000)-t.

A vizsgált problémában minden nem-egyelemű időablakkal rendelkező j munkára minden i gépen meghatározom a $\tilde{LB}_{ijt}$ értéket ($\boldsymbol{\lambda}^*$ használatával) úgy, hogy j munka kezdését t ($e_j \leq t \leq d_j - p_j$) időpontban rögzítem. Az eredmények megfelelő feldolgozásával egyes munkák időablaka szűkíthető lehet megoldás elvesztése nélkül, továbbá az alsó korlát is javulhat.

2.2. Tétel. *Az* $\tilde{LB}_i = \max_{j \in J_i} \min_{e_j \leq t \leq d_j - p_j} \tilde{LB}_{ijt}$ *érték érvényes alsó korlátot ad a keresési fa megfelelő csúcsához tartozó feladatra.*

2.3. Tétel. *Amennyiben a shaving eljárás a j munka időablakából kizárja a t időpontot, akkor nem létezhet a keresési fa megfelelő csúcsához tartozó feladatban olyan, az aktuálisan ismert legjobb megoldásnál jobb ütemterv, ahol j munka a t időpontban kezdődik.*

2.4.4. Felső korlát számítása

A korlátozás és elágazás algoritmus minden csúcsában az aktuális időablakok szerinti megengedett megoldást keresek. Mivel az egyes gépek ütemtervének megengedettsége független az ütemterv maradék részétől, megtehető, hogy a gépeken egyesével, külön-külön készítsem el az ütemtervet. Elfogadható számítási időben erre csak heurisztikák vagy approximációs algoritmusok segítségével van lehetőség, hiszen az egygépes feladatokban a megengedett ütemterv létezése NP-nehéz kérdés. Egy adott gépen a kiinduló ütemtervet azon ötlet alapján készítem el, hogy egy jó megoldás várhatóan hasonlítani fog a Lagrange-relaxációból meghatározható tervhez. Ha nem sikerül ily módon megengedett ütemtervet találni, akkor egy approximációs algoritmust alkalmazok. Amennyiben valamelyik módszer sikerrel járt, akkor a kiinduló megoldást egy heurisztikus algoritmus segítségével megpróbálom tovább javítani.

2.1. Lemma. *Egy adott gépre Hall és Shmoys (1982) approximációs algoritmusa segítségével a következő eredmények egyikére juthatunk:*

(a) Létezik megengedett megoldás és ezt meg is találtuk.

(b) Létezhet megengedett megoldás, de nem találtunk ilyet.

(c) Nem létezhet megengedett megoldás.

2.2. Algoritmus. *Az egyes gépeken egy megengedett megoldás keresése a következő lépések szerint történik:*

1. *Amennyiben a Lagrange-relaxáció adott egy megengedett megoldást, akkor használjuk ezt változtatás nélkül.*

2. *Amennyiben a Lagrange-relaxáció által adott megoldás tört értékeket tartalmaz, próbáljunk ebből megengedett megoldást készíteni kerekítésekkel. Ha sikerül, akkor használjuk ezt a megoldást.*

3. *Ha az előző lépésekben nem sikerült megengedett megoldást találni, akkor használjuk a 2.1. lemmát. Ennek segítségével a következő három eset lehetséges:*

   (a) *Találtunk megengedett megoldást. Ekkor használjuk ezt.*

   (b) *Létezhet megengedett megoldás, de nem találtunk ilyet. Ebben az esetben nem számolunk felső korlátot.*

   (c) *Nem létezik megengedett megoldás, következésképp a keresési fa megfelelő csúcsát elvethetjük.*

A kiinduló megoldást egy lokális kereséssel javítom azokon a gépeken, amelyeken a megoldást a fenti algoritmus 2. vagy 3. lépése szolgáltatta. A keresés során a (Drótos és Kis; 2011) cikkben szereplő, polinomiális algoritmust használom.

2.4.5. Elágazás

A keresési fa minden csúcsában előre meghatározott számú gyereket generálok. Ez azt jelenti, hogy egy kiválasztott j munka időablakát B folytonos részre particionálom, és az adott munka időablaka az egyes gyerekekhez tartozó feladatokban a megfelelő partíció lesz. Mivel azt szeretném elérni, hogy az egyes időablakokban a 2.4.3. szakaszban definiált $\tilde{LB}_{ijt}$ értékek hasonlók legyenek, a particionálást egy megfelelően felírt optimalizálási problémaként oldom meg.

Ezzel a módszerrel az időablakokat „természetes” módon tudom felosztani (az egyes gyerekek a kiválasztott munka „kívánatos” és „nem kívánatos” időablakaihoz tartozó részproblémáknak fognak megfelelni). A bemutatott elágazási séma azt is biztosítja, hogy a testvér csúcsok alsó korlátja várhatóan különbözni fog, így az elágazás és korlátozás algoritmus az ígéretes részfeladatokat fogja először kiértékelni.

2.4.6. Számítási eredmények

A bemutatott egzakt és heurisztikus módszereket C++ nyelven implementáltam, és véletlenszerűen generált feladatokat használtam a tesztelésükre. Az egyes tesztpéldányokat felírtam az (1)-(5) által definiált IP problémákként is, ezekre futtattam az ILOG CPLEX 11.2 általános MIP-megoldó programot, és az így kapott eredményeket összehasonlítottam a saját algoritmusom eredményeivel. A tesztproblémákban az egyes feladatok $m = 5, 10, 20$ gépet, valamint gépenként $t = 10, 15$ és 20 munkát tartalmaznak. Az egyes (m, t) teszthalmazokban egyenként 10 véletlenszerűen generált feladat szerepel.

A 2. és 3. táblázat a kiszámított felső- és alsó korlát arányát ($UB/LB - 1$, százalékban kifejezve) mutatja teszthalmazonként átlagolva lineáris és négyzetes célfüggvény esetén. A BB-vel jelölt oszlopokban a jelen tézisben bemutatott módszer eredményei, míg a CPX oszlopban az ILOG CPLEX eredményei szerepelnek. A felhasznált CPU-idő limitje minden futtatásnál 1800 másodperc volt. Négyzetes célfüggvény esetén a CPLEX csak a legkisebb tesztpéldákra tudott akár alsó, akár felső korlátot adni a megadott időkorláton belül.

A teszteredményekből jól látható, hogy az általam kidolgozott algoritmus lineáris célfüggvény használata esetén jobban teljesít a CPLEX-nél nagyobb méretű problémákon, négyzetes célfüggvényt használva pedig továbbra is bizonyítottan jó minőségű megoldásokat ad, ezzel szemben a CPLEX nehézségekbe ütközik akár alsó-, akár felső korlátok kiszámításában.

2.5. Publikációk

A 2. tézis eredményeit a (Drótos és Kis; 2011; Drótos et al.; 2007b; Drótos és Kis; 2008, 2010b) publikációkban tettem közzé.

	m5		**m10**		**m20**		**avg**	
	BB	*CPX*	*BB*	*CPX*	*BB*	*CPX*	*BB*	*CPX*
t10	6.16%	3.10%	0.74%	0.24%	0.36%	0.37%	**2.42%**	**1.24%**
t15	12.94%	11.28%	5.08%	5.96%	0.41%	1.62%	**6.14%**	**6.29%**
t20	18.39%	17.15%	5.41%	7.48%	2.19%	10.39%	**8.66%**	**11.67%**
avg	**12.49%**	**10.51%**	**3.74%**	**4.56%**	**0.99%**	**4.13%**	**5.74%**	**6.40%**

2. táblázat. Felső- és alsó korlát aránya átlagosan, lineáris célfüggvény esetén, $C_\ell = \lfloor \sum_{j \in J} b_{\ell j} p_j / D \rfloor$, 3 erőforrástípussal.

	m5		**m10**		**m20**		**avg**	
	BB	*CPX*	*BB*	*CPX*	*BB*	*CPX*	*BB*	*CPX*
t10	2.31%	1.51%	0.85%	–	0.24%	–	**1.13%**	–
t15	3.77%	–	1.20%	–	0.60%	–	**1.86%**	–
t20	4.31%	–	2.71%	–	0.37%	–	**2.46%**	–
avg	**3.46%**	–	**1.59%**	–	**0.40%**	–	**1.82%**	–

3. táblázat. Felső- és alsó korlát aránya átlagosan, négyzetes célfüggvény esetén, $C_\ell = 0$, 3 erőforrástípussal.

3. Komplex ipari ütemezési problémák megoldására szolgáló keretrendszer bemutatása

3.1. Motiváció

Bár a gépütemezés egy behatóan tanulmányozott témakör széles irodalommal, a legtöbb megközelítés egyszerűsített modellekkel dolgozik. Az ilyen egyszerűbb, relaxált modellek előnye az, hogy jól tükrözik a megoldandó probléma lényegi szerkezetét a kevésbé érdekes technikai részletek elhanyagolása mellett, de valós ipari környezetben nem mindig használhatók. Ezért tartottam fontosnak olyan módszerek kidolgozását, amelyek általánosan használhatók a gyakorlatban, változatos ütemezési problémákra.

A célom egy olyan testreszabható, moduláris keretrendszer kidolgozása volt, ami job-shop jellegű problémák megoldására használható úgy, hogy képes figyelembe venni a gyakorlatban előforduló legtöbb korláttípust (korlátos megújuló és nem megújuló erőforrások, gép- és sorrendfüggő átállási idők, termékhierarchiák, stb.), továbbá egyszerre többféle célfüggvény optimalizálására is alkalmas. Az is fontos szempont volt, hogy a megoldások tetszetősek legyenek az emberi döntéshozók számára is, vagyis az automatikusan készített ütemtervek jobbak legyenek, mint a kézzel készítettek.

3.2. Kapcsolódó eredmények

Egy ütemezőrendszer nem kizárólag az adott probléma megoldását végző algoritmusokból áll, hanem integráns részét képezi az adott cég üzleti információs rendszerének. Ennek következtében fontos, hogy kövessük az egyedi termelésütemező rendszerek fejlesztése és integrációja során az ismert ajánlásokat. Framinan és Ruiz (2010) egy átfogó elemzést készített az ilyen rendszerekkel szemben támasztott követelményekről a menedzsment szinttől az ütemezésig.

A job-shop ütemezés jól ismert és régóta tanulmányozott probléma, ráadásul sok eredmény született az olyan típusú kiegészítéseiről is, amilyeneket nekem is célom kezelni; lásd pl. (Schutten; 1998; Mason et al.; 2005; Mönch et al.; 2007). A kereskedelmi forgalomban kapható ütemező szoftverek (pl. SAP-hoz készült ütemező szoftverek) tipikusan nem képesek kielégíteni a valós ipari gépütemezési igényeket. Ennek több oka is van: (1) a megújuló és nem megújuló erőforrások és a gépek által jelentett korlátok kezelése nem lehetséges egyszerre; (2) sokszor csak nagyon korlátozott modelleket képesek kezelni (pl. az egyes gépcsoportok csak ekvivalens gépeket tartalmazhatnak, stb.). A tudományos publikációk-

ban szereplő algoritmusok sokszor csak specifikus, jól definiált problémákon működnek, és ennek következtében sokszor nem jól adaptálhatók más modellekre.

Az ebben a tézisben bemutatott eredményeim nagyban támaszkodnak heurisztikus algoritmusokra. Ezekről ad általános áttekintést Silver (2004), aki érveket is hoz az ilyen módszerek gyakorlati alkalmazásának hasznosságára kombinatorikus optimalizálási problémák esetén.

3.3. Eredmények

Jelen tézisben egy több fázisú keretrendszert mutatok be, ami sokféle ütemezési probléma kezelésére használható. A keretrendszerbe az aktuális probléma típusától függően lehet különböző algoritmusokat beilleszteni. Az egyes fázisokban egyre több korlátot veszek hozzá az adott problémához, ezzel párhuzamosan egyre több paramétert rögzítek. Ez lehetővé teszi, hogy a különböző célfüggvények optimalizálása az arra leginkább célszerű algoritmusokkal történjék, miközben az előző fázisokban meghozott döntéseket is figyelembe veszik.

3.1. Algoritmus. *A komplex ipari ütemezési problémákra javasolt keretrendszer a következő fázisokból áll:*

1. *Alapvető döntések meghozatala, valamint alsó korlát számítása*
2. *A legfontosabb szempontok szerint optimalizált ütemterv készítése*
3. *Az ütemterv finomhangolása az összes követelmény teljesítésének érdekében*

3.2. Algoritmus. *Algoritmus nem megújuló erőforrások (pl. alapanyagok) kiosztására prioritással rendelkező munkák esetén.*

3.3. Algoritmus. *Lokális keresésekben használható algoritmus a munkák kezdési és befejezési időpontjainak gyors újraszámítására job shop jellegű ütemezési problémákban.*

3.4. Algoritmus. *Algoritmus megújuló erőforrások kiosztására olyan ütemezési problémákban, ahol egy, a helyszínek által meghatározott hierarchiát is figyelembe kell venni.*

3.5. Algoritmus. *Algoritmus megújuló erőforrások által jelentett korlátok kielégítésére olyan környezetben, ahol a konfliktusok gépek műszakjainak kikapcsolásával oldhatók fel.*

A javasolt keretrendszer implementációi telepítve lettek a nagykanizsai GE Hungary Zrt. – Consumer & Industrial, valamint az egri Bosch Rexroth Pneumatika Kft. gyárakban.

3.3.1. I. fázis: Alapvető döntések meghozatala és alsó korlát számítása

A gyakorlati problémák jelentős részében vannak olyan döntések, amik akár a megoldandó ütemezési probléma alapadataira is hatással lehetnek. Tipikus példa erre egy olyan feladat, ahol az egyes munkák több alternatív módon gyárthatók, és az alternatíva megválasztása határozza meg azt, hogy milyen műveleteket kell az adott munkán elvégezni. Sorozatos rossz döntések esetén előállhatna olyan helyzet, hogy a megtalált megoldás nem elég jó minőségű, továbbá kis változtatásokkal nem lehet rajta javítani, viszont más döntések kipróbálása algoritmikus szempontból túl komplex lenne.

Sok esetben viszont lehetőség van készíteni egy, a standard megoldók által is kezelhető relaxált matematikai modellt. Egy ilyen relaxált felírás akár az ütemezés szempontjából nagyon fontos paramétereket is elhanyagolhat: például egy konkrét alkalmazásban ilyenek a megelőzési korlátok, átállási idők, stb. Ennek a megközelítésnek két nagy előnye van:

- a legmagasabb szintű döntések egy optimalizációs eljárás segítségével születnek meg, tehát biztonsággal rögzíthetők a későbbi fázisokban használt algoritmusok számára;
- a relaxált probléma megoldása alsó korlátot ad a célfüggvény(ek)re, aminek segítségével mérhető a végső megoldás jósága.

Az alapanyagkiosztás szintén olyan terület, ahol biztonsággal véglegesíthetők a kezdeti döntések. Az olyan problémák, ahol egyszerre kell a nem megújuló erőforrásokat kiosztani és egy komplex gépmodellt (pl. flow shop vagy job shop) is kezelni, sokszor túl nehéznek bizonyulnak a direkt módon megoldáshoz. Ezt a nehézséget úgy hidalom át, hogy az első fázisban végzem el a nem megújuló erőforrások kiosztását. Erre egy olyan algoritmust dolgoztam ki, ami az iparban előforduló tipikus problématípusokra ad jó megoldást. Mivel a legtöbb környezetben az anyagok beérkezése egy előre meghatározott, adott periódusban nem megváltoztatható terv szerint történik, egy ütemterv csak akkor megengedett, ha nem okoz anyaghiányt ebben a periódusban (későbbi időpontokra már megrendelhető a szükséges mennyiségű alapanyag). A 3.2. algoritmus az egyes munkáknak olyan legkorábbi kezdési időpontokat határoz meg, hogy az ezeket betartó ütemtervekben nem léphet fel anyaghiány.

3.3.2. II. fázis: A legfontosabb szempontok szerint optimalizált ütemterv készítése

Az előző fázisban meghozott döntések alapján már elkészíthető egy kezdeti, megengedett ütemterv. Az egyik legígéretesebb megközelítés ennek javítására egy irányított lokális

keresés használata, mint például a tabukeresés. A tabukeresés nagy előnye, hogy a futása során végig megengedett ütemtervekkel dolgozunk, valamint a végeredmény lokálisan optimális lesz. Ez azt jelenti, hogy kis változtatásokkal nem lehetséges a megtalált ütemterven javítani, vagyis az ütemező rendszer felhasználói számára is meggyőző megoldásokat ad. Ezen felül sokféle korlátozás kikényszeríthető úgy, hogy ehhez az algoritmust csak kismértékben kell módosítani.

A szomszédság megfelelő megválasztásával és az adott probléma struktúrájának kihasználásával a keresés felgyorsítható. A 3.3. algoritmus ezt használja ki: ha egy ütemtervben egy változtatás csak a terv kis részére van hatással, akkor elég csak ezt a részt kiértékelni, a maradék részben pedig felhasználhatjuk a korábbi kiértékelések eredményét. Munkámban bemutatok még többféle keresési stratégiát és ezek összehasonlítását is késés jellegű célfüggvények esetén.

Amennyiben a megújuló erőforrásokat komplex módon szükséges kezelni (pl. adott a helyszíneknek egy hierarchiája, ami befolyásolja az erőforráskiosztást), akkor a 3.3. algoritmus nem használható, mivel olyan munkák is lehetnek konfliktusban, amik között egyébként nincs kapcsolat a precedenciák szerint. Az ilyen feladatok kezelésére kidolgoztam egy általános modellt, és adtam egy algoritmust (3.4. algoritmus), ami hatékonyan kiszámítja egy ütemtervben a munkák kezdési és befejezési időpontját ebben a modellben.

3.3.3. III. fázis: Az ütemterv finomhangolása az összes követelmény teljesítésének érdekében

Ebben a fázisban már adott egy, a fő ütemezési szempontok szerint jó minőségű ütemterv, amely azonban lehetséges, hogy tovább javítható egyéb szempontok szerint úgy, hogy közben nem romlik szignifikánsan a már vizsgált célfüggvények szerint. Egy tipikus ilyen másodlagos célfüggvény lehet például az átállási idők valamely függvényének minimalizálása a gépeken, hiszen a gyakori átállások magas gyártási költségeket jelentenek, de jellemzően ennek minimalizálása kevésbé fontos, mint a határidők betartása.

Ennek a fázisnak a fő ötlete az, hogy a munkák gépekhez rendelését rögzítem, az elsődleges célfüggvény romlására bevezetek egy toleranciát, és ezen korlátok mellett optimalizálom tovább az ütemtervet. Erre az optimalizálásra az előző fázisban használt lokális keresés algoritmusok gyakran felhasználhatók kisebb módosításokkal.

Ezentúl bizonyos egyszerűbb erőforrásmodellek esetén elég lehet a megújuló erőforrások által jelentett korlátok kezelése ebben a fázisban. Egy ilyen, a gyakorlatban is előforduló modell előírhatja azt, hogy erőforráshiány esetén csak teljes műszakok kikapcsolására van

lehetőség az egyes gépeken. A 3.5. algoritmust erre az esetre dolgoztam ki. Ez az előző fázisokban használt eszközök segítségével számít korlátokat, majd ezek használatával iteratívan készít kisméretű, az általános megoldók által hatékonyan kezelhető egészértékű programozási problémákat. Ezen problémák megoldása mutatja meg, hogy mely műszakokat kell mely gépeken kikapcsolni ahhoz, hogy az ütemterv kielégítse az erőforráskorlátokat.

3.3.4. Számítási eredmények

A bemutatott módszereket teszteltem mind véletlenszerűen generált, mind ipari adatokból készült feladatok segítségével. A vizsgált problémák átlagosan ∼130 gépet és ∼3500 munkát tartalmaztak. A felső és alsó korlát aránya összkésés elsődleges célfüggvény esetén átlagosan 1.29 volt a legjobbnak bizonyuló keresési stratégia használatával, miközben a többi célfüggvény értéke is közel volt az optimálishoz.

3.4. Publikációk

A 3. tézis eredményeit a (Drótos et al.; 2009; Kis et al.; 2006; Drótos et al.; 2007a; Drótos és Kis; 2010a; Monostori et al.; 2008) publikációkban tettem közzé. A bemutatott megoldások egyes elemei részei a (Nakano et al.; 2012 – Japan és 2013 – WIPO) szabadalomnak.

Hivatkozások

Błażewicz, J. (2001). *Scheduling Computer and Manufacturing Processes*, 2^{nd} edn, Springer-Verlag, Berlin.
`http://www.google.hu/books?id=0rrq1xlJ1dsC`

Błażewicz, J., Lenstra, J. és Rinnooy Kan, A. (1983). Scheduling subject to resource constraints: classification and complexity, *Discrete Applied Mathematics* **5**(1): 11 – 24.
`http://www.sciencedirect.com/science/article/pii/0166218X83900124`

Boysen, N., Bock, S. és Fliedner, M. (2013). Scheduling of inventory releasing jobs to satisfy time-varying demand: an analysis of complexity, *Journal of Scheduling* **16**(2): 185–198.
`http://dx.doi.org/10.1007/s10951-012-0266-0`

Briskorn, D., Choi, B.-C., Lee, K., Leung, J. és Pinedo, M. (2010). Complexity of single machine scheduling subject to nonnegative inventory constraints, *European Journal of Operational Research* **207**(2): 605 – 619.
`http://www.sciencedirect.com/science/article/pii/S0377221710003929`

Burgess, A. R. és Killebrew, J. B. (1962). Variation of activity level on a cyclic arrow diagram, *Journal of Industrial Engineering* **13**: 76–83.

Carlier, J. és Pinson, E. (1994). Adjustment of heads and tails for the job-shop problem, *European Journal of Operational Research* **78**(2): 146 – 161.
`http://www.sciencedirect.com/science/article/pii/0377221794903794`

Demeulemeester, E. és Herroelen, W. (2002). *Project Scheduling: A Research Handbook*, International Series in Operations Research & Management Science, Kluwer Academic Publishers, Boston/Dordrecht/London.
`http://www.google.hu/books?id=pHetPl2LOYgC`

Framinan, J. M. és Ruiz, R. (2010). Architecture of manufacturing scheduling systems: Literature review and an integrated proposal, *European Journal of Operational Research* **205**(2): 237 – 246.
`http://www.sciencedirect.com/science/article/pii/S0377221709006419`

Garey, M. R. és Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco.

Grigoriev, A., Holthuijsen, M. és van de Klundert, J. (2005). Basic scheduling problems with raw material constraints, *Naval Research Logistics (NRL)* **52**(6): 527–535.
`http://dx.doi.org/10.1002/nav.20095`

Hall, L. A. és Shmoys, D. B. (1982). Jackson's rule for single-machine scheduling: making a good heuristic better, *Mathematics of Operations Research* **17**: 22–35.

Hartmann, S. és Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem, *European Journal of Operational Research* **207**(1): 1 – 14.
http://www.sciencedirect.com/science/article/pii/S0377221709008558

Kellerer, H., Kotov, V., Rendl, F. és Woeginger, G. J. (1998). The stock size problem, *Operations Research* **46**(3): S1–S12.
http://www.jstor.org/stable/3840850

Martin, P. és Shmoys, D. (1996). A new approach to computing optimal schedules for the job-shop scheduling problem, *in* W. Cunningham, S. McCormick és M. Queyranne (eds), *Integer Programming and Combinatorial Optimization*, Vol. 1084 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 389–403.
http://dx.doi.org/10.1007/3-540-61310-2_29

Mason, S., Fowler, J., Carlyle, W. és Montgomery, D. (2005). Heuristics for minimizing total weighted tardiness in complex job shops, *International Journal of Production Research* **43**(10): 1943–1963.
http://www.tandfonline.com/doi/abs/10.1080/00207540412331331399

Mönch, L., Schabacker, R., Pabst, D. és Fowler, J. W. (2007). Genetic algorithm-based subproblem solution procedures for a modified shifting bottleneck heuristic for complex job shops, *European Journal of Operational Research* **177**(3): 2100 – 2118.
http://www.sciencedirect.com/science/article/pii/S0377221705008532

Schutten, J. (1998). Practical job shop scheduling, *Annals of Operations Research* **83**: 161–178. 10.1023/A:1018955929512.
http://dx.doi.org/10.1023/A:1018955929512

Silver, E. A. (2004). An overview of heuristic solution methods, *The Journal of the Operational Research Society* **55**(9): 936–956.
http://www.jstor.org/stable/4101930

Torres, P. és Lopez, P. (2000). Overview and possible extensions of shaving techniques for job-shop problems, *CP-AI-OR '2000*, pp. 181–186.

Saját publikációk

Publikációk nemzetközi folyóiratokban

Drótos, M., Erdős, G. és Kis, T. (2009). Computing lower and upper bounds for a large-scale industrial job shop scheduling problem, *European Journal of Operational Research* **197**(1): 296 – 306.
http://www.sciencedirect.com/science/article/pii/S0377221708004700

Drótos, M. és Kis, T. (2011). Resource leveling in a machine environment, *European Journal of Operational Research* **212**(1): 12 – 21.
http://www.sciencedirect.com/science/article/pii/S037722171100097X

Drótos, M. és Kis, T. (2013a). Scheduling of inventory releasing jobs to minimize a regular objective function of delivery times, *Journal of Scheduling* **16**(3): 337–346.
http://link.springer.com/article/10.1007%2Fs10951-012-0297-6

Publikációk hazai folyóiratokban

Monostori, L., Váncza, J., Kis, T., Kádár, B., Karnok, D., Drótos, M. és Egri, P. (2008). Valós időben együttműködő vállalatok: egy ipari-akadémiai projekt eredményei, *Gépgyártás* **48**(4): 11 – 15.

Publikációk konferenciakiadványokban

Drótos, M., Erdős, G. és Kis, T. (2007a). A lower bound and a tabu search algorithm for scheduling batches in a proportional job shop, *MAPSP 2007, 8th Workshop on Models and Algorithms for Planning and Scheduling Problems*, Istanbul.

Drótos, M., Erdős, G. és Kis, T. (2007b). New techniques to handle workforce constraints in parallel machine scheduling, *MIM'07: IFAC workshop on manufacturing modelling*, Budapest, pp. 101–105.

Drótos, M., Györgyi, P. és Kis, T. (2013). Approximation algorithms for machine scheduling with non-renewable resources, *MAPSP 2013: 11th Workshop on Models and Algorithms for Planning and Scheduling Problems*, Pont à Mousson.

Drótos, M. és Kis, T. (2008). Minimize overtime in a parallel machine environment, *VOCAL 2008: Veszprém Optimization Conference: Advanced Algorithms*, Veszprém.

Drótos, M. és Kis, T. (2010a). Challenges in planning and scheduling in discrete manufacturing and assembly, *Factory Automation 2010*, Kecskemét.

Drótos, M. és Kis, T. (2010b). Resource levelling in a parallel machine environment, *PMS2010: 12th International Conference on Project Management and Scheduling*, Tours, pp. 171–174.

Drótos, M. és Kis, T. (2013b). Scheduling of inventory releasing jobs, *EURO2013: 26th European Conference on Operational Research*, Rome.

Kis, T., Drótos, M., Erdős, G. és Kovács, A. (2006). The interval flow shop model, *PMS2006: Tenth International Conference on Project Management and Scheduling*, Poznan, pp. 217–221.

Szabadalmak

Nakano, T., Nonaka, Y., Drótos, M., Erdős, G., Kis, T., Kovács, A., Monostori, L. és Váncza, J. (2012 – Japan és 2013 – WIPO). Processing step planning device. App. number: 2012-054166 (Japan), PPCT/JP2013/056549 (WIPO).
`http://patentscope.wipo.int/search/en/WO2013137159`

Egyéb publikációk

Drótos, M. (2012). *Algoritmuselmélet (BMEVISZA213) feladatgyűjtemény.*
`http://www.cs.bme.hu/~drotos/algfgy.pdf`