Eötvös Loránd Tudományegyetem

Természettudományi Kar

Farkas Dóra

Ütemezési problémák speciális ütközési feltételekkel

MSc szakdolgozat

Témavezető:

Király Tamás
Operációkutatási Tanszék

Budapest, 2020

NYILATKOZAT

Név: Farkas Dóra

ELTE Természettudományi Kar, szak: Alkalmazott matematikus

NEPTUN azonosító: DN788L

Szakdolgozat címe:
Ütemezési problémák speciális ütközési feltételekkel

A **szakdolgozat** szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló szellemi alkotásom, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2020. 05. 30.

Farkas Dóra

a hallgató aláírása

Köszönetnyilvánítás

Köszönettel tartozom Király Tamásnak, aki tanulmányaim során végig vállalta a témavezetői feladatokat. Észrevételeivel és javaslataival folyamatosan támogatta a munkám, és segítette, hogy ez a szakdolgozat létrejöhessen.

Köszönöm továbbá a családomnak és szeretteimnek, hogy az egyetemi éveim alatt is végig mellettem álltak.

Tartalomjegyzék

1. Bevezetés **5**

2. Ütemezési feladatok **7**
2.1. Ütközési feltételek 8
2.2. Job shop 8

3. Bonyolultság **10**
3.1. Polinomiális algoritmus 2 gépre 10
3.1.1. Dinamikus programozás 10
3.1.2. Geometriai megoldás 12
3.2. NP-nehézség 3 gépre 14
3.3. Job shop NP-nehézsége 3 gép és 3 munka esetén 21

4. Közelítő algoritmusok **23**
4.1. 2-approximáció 23
4.2. 5/3-approximáció 3 gépre 24
4.3. 11/6-approximáció 4 gépre 26

5. Összegzés **33**

Irodalomjegyzék **35**

1. fejezet

Bevezetés

A dolgozatban olyan dedikált gépes ütemezési feladatokról lesz szó, melyek feltételei között ütközési korlátok szerepelnek. A szükséges fogalmak bevezetése után a feladatokat először bonyolultságelméleti szempontból vizsgáljuk meg. Ha 2 gép van, akkor az optimális megoldás polinomiális időben meghatározható, erre két algoritmust is bemutatok. A feladat 3 gépes változatáról viszont sikerült bebizonyítanom, hogy NP-nehéz. Emiatt a 3 vagy több gépet tartalmazó problémákra közelítő algoritmusokat dolgoztam ki.

Mielőtt az ütemezési feladatok részletes tárgyalására rátérnénk, ismerkedjünk meg a többfejes 3D épületnyomtatással, amely a motivációt adta az ütemezési problémák vizsgálatához. Ennek leírásához elsősorban Zhang és Khoshnevis [1] cikkét használjuk fel.

3D nyomtatás segítségével teljesen automatizált módon alakíthatjuk ki a kívánt szerkezeteket. A nyomtatás megkezdése előtt létre kell hoznunk a szerkezet számítógépes modelljét. Majd azt párhuzamos síkokkal rétegekre kell szeletelni, mivel a nyomtatás során a gép rétegről rétegre haladva építi fel a tárgyat. Ezután minden rétegben meg kell határozni, hogy a nyomtatófej milyen útvonalat járjon be, közben mikor milyen sebességgel haladjon, és mekkora legyen az anyag áramlási sebessége. Többek között a réteg vastagsága, vagy az útvonalban található éles sarkok nagyban befolyásolják a kész termék felületének minőségét [2]. Ha ez nem megfelelő, akkor a nyomtatást követően további utómunkára is szükség van.

A 3D épületnyomtatás során a nyomtató gép az épület falait nyomtatja ki a megfelelő építőanyagból. Az építkezésnél használt hagyományos módszerekkel szemben ezzel a technológiával gyorsabban, olcsóbban és biztonságosabban állíthatunk fel egyedi tervezésű házakat. Az épületnyomtatás különösen hasznos lehet katasztrófa sújtotta területek gyors helyreállításában, de pl. a Holdon is fel lehetne használni különböző szerkezetek megépítéséhez [3].

A nyomtatófejek útvonalának optimális megválasztása különösen fontos a hatékonyság szempontjából, pl. a nyomtatási idő csökkentésében. Az általunk vizsgált feladatban az ajtók és ablakok helyét leszámítva a rétegek megegyeznek, ezért elég egyetlen réteggel foglalkozni. A nyomtatófej útvonaláról feltesszük, hogy ugyanabban a pontban fejeződik be, mint ahol az adott réteg nyomtatását kezdte, így a fej feljebb emelését követően rögtön kezdhetjük a következő réteg nyomtatását. Azt is feltesszük, hogy a beton áramlását tökéletesen szabályozni lehet, és így a nyomtatófejet tetszőleges pontban megállíthatjuk, ill. újraindíthatjuk. Az útvonal meghatározásához az épület modelljének egy rétegét reprezentálhatjuk egy gráffal. Ebben egy olyan Hamilton-kört kell keresnünk, amely tartalmazza a falrészekhez tartozó éleket. Az élek költségét a nyomtatás különböző költségeiből számolhatjuk ki. Többek között figyelembe kell vennünk, hogy mennyi ideig tart, amíg a fej

egy falrész nyomtatását követően a következő falrészhez eljut. Ezt a két pont távolságából és a fej sebességéből számolhatjuk ki. Ezen kívül a nyomtatófej elfordításához szükséges időt is meg kell határoznunk. Az így kapott gráfban az utazóügynök feladat megoldásával meghatározhatjuk a nyomtatófej útvonalát. Erre használhatjuk Lin és Kernighan heurisztikus algoritmusát [4], illetve ennek egy továbbfejlesztett változatát.

Nagyobb épületek esetén több nyomtatófejre is szükség lehet. Mivel, ha egy réteg nyomtatása túl sokáig tart, azaz túl sok idő telik el, amíg a nyomtatófej ugyanarra pontra visszaér a következő rétegben, akkor a szomszédos rétegek nem tudnak megfelelően egymáshoz tapadni. Több nyomtatófej használatával időt takaríthatunk meg, mert a fejek egymás mellett, párhuzamosan dolgozhatnak. Azonban ilyenkor arra is figyelnünk kell az útvonalak meghatározásakor, hogy a fejek nehogy összeütközzenek. A nyomtatás előkészítése során meg kell adnunk, hogy mely falrészeket melyik fej fogja kinyomtatni. Azért, hogy a fejek nagyjából egyszerre végezzenek az adott réteg nyomtatásával, a munkát egyenletesen kell szétosztani közöttük. Ehhez először a nyomtatási területet egymástól egyenlő távolságra lévő párhuzamos egyenesekkel vágjuk fel, és minden fej útvonalát külön-külön meghatározzuk az egyfejes nyomtatóknál használt módszerhez hasonlóan. Ha az így kapott nyomtatási idők között túl nagy a különbség, akkor a nyomtatási területek határát megváltoztatjuk, és újra meghatározzuk az útvonalakat. Ezt addig ismételjük, amíg nyomtatási idők eltérése a megfelelő érték alá nem kerül.

Ha az egyik fej a saját területének szélén helyezkedik el, akkor a tartó állványzat átlóghat a szomszédos fej területére, és ilyenkor előfordulhat, hogy a fejek összeütköznek egymással. Ennek elkerülésére több különböző módszer is lehetséges, melyek kombinálhatók is egymással.

Ha a fejek területének közös határánál megfelelő szélességű ütköző zónákat alakítunk ki, és elérjük, hogy ebben egyszerre csak egy fej tartózkodjon, akkor ezzel elkerüljük az ütközést. Minden fejre határozzuk meg azokat a pontokat, amikor a fej belép az ütköző zónába, vagy kilép onnan. Ezekkel a nyomtatás összidejét ütköző és nem ütköző szakaszokra bonthatjuk. Ha ezen intervallumok közé megfelelő hosszú várakozási időket szúrunk be, akkor elérhetjük, hogy az azonos ütköző zónához tartozó szakaszok diszjunktak legyenek. Várakozás esetén a nyomtatófejet az adott pontban megállítjuk, és csak a várakozási idő letelte után folytathatja útját az eredeti útvonalterv szerint. A várakozások helyének és hosszának meghatározásakor az a célunk, hogy a nyomtatási összideje minimális legyen, ez egy ütemezési feladat megoldásához vezet. A későbbiekben ezt a problémát vizsgáljuk meg részletesen.

Ha felhasználjuk, hogy a fejek zárt útvonalat járnak be, és megváltoztatjuk ennek kezdőpontját, akkor így elforgathatjuk az útvonalat. Ha ezután a két szomszédos fej távolsága minden időpillanatban elegendően nagy, akkor várakozások beszúrása, a nyomtatási idő megnövelése nélkül is elérhetjük az ütközésmentességet. Ez a módszer azonban nem mindig használható.

Egy másik lehetőség, hogy 1-1 fej területét további részekre osztjuk. Ekkor minden fejhez több útvonal is tartozik, így az útvonal hossza és ezzel a nyomtatási idő megnő. De ha ezeket a részútvonalakat megfelelő sorrendben járják be, akkor ezzel is elkerülhető az ütközés.

2. fejezet

Ütemezési feladatok

Jordán Tibor jegyzete [5] alapján bemutatjuk az ütemezési feladatokhoz kapcsolódó alapfogalmakat.

Az ütemezéselméletben olyan problémákat vizsgálunk, ahol az a feladatunk, hogy munkákat ütemezzünk gépeken a megadott feltételek mellett. Meg kell adnunk, hogy melyik munkát melyik gépen és mikor végezzük el. A megengedett ütemtervek közül egy olyat kell kiválasztani, amely az adott célfüggvényt minimalizálja.

Általános szabály, hogy minden gép egy időben legfeljebb csak egy munkán dolgozhat. Alapesetben a munkákat nem lehet megszakítani. Azonban, ha ez mégis megengedett, akkor egy munka darabjait különböző gépeken is elvégezhetjük, de ekkor is teljesülnie kell, hogy minden munkán egyszerre csak egy gép dolgozhat.

Az ütemezési feladatokban adott n db munka és m db gép. Az i-edik munka megmunkálási idejét $p(i)$-vel jelöljük. A később vizsgált feladatokban minden munka további műveletekre bomlik, ekkor az i-edik munka j-edik műveletének megmunkálási idejét $p(i,j)$-vel jelöljük.

Egy adott T ütemterv esetén az i-edik munka, ill. ennek j-edik műveletének kezdési és befejezési idejét $S^T(i)$ és $C^T(i)$, ill. $S^T(i,j)$ és $C^T(i,j)$ jelöli. Megszakítás nélküli ütemtervekben ezekre teljesül, hogy $S^T(i)+p(i) = C^T(i)$, $S^T(i,j)+p(i,j) = C^T(i,j)$. Ha egyértelmű, hogy melyik ütemtervről van szó, akkor a T jelölését elhagyjuk. Az optimális megoldást mindig $*$-gal jelöljük.

Ha több egyforma gép van, akkor párhuzamosnak mondjuk őket. Ha a műveletek előre hozzá vannak rendelve a gépekhez, azaz az inputban meg van adva, hogy az i-edik munka j-edik műveletét a $\mu(i,j)$ gépen kell elvégezni, akkor dedikált gépes feladatról beszélünk.

A feltételek között lehetnek megelőzési korlátok. Ekkor a munkák halmazán adott egy részben rendezés, és egy munkát csak akkor lehet elkezdeni, ha az őt megelőző összes munkát már befejeztük. Ennek egy speciális esete, ha a megelőzési feltételek diszjunkt láncokból állnak, ezt *chains*-el jelöljük.

A lehetséges célfüggvények közül a későbbiekben mi a teljes átfutási időt, azaz $C_{max} = \max\{C(i) : i = 1, \dots n\}$-t fogjuk minimalizálni. De gyakran használják az átlagos átfutási időt ($\frac{1}{n}\sum C(i)$), a súlyozott átlagos átfutási időt ($\frac{1}{n}\sum w(i)C(i)$), vagy ha a műveletek határideje is meg van adva, akkor a maximális késést.

Ütemezési feladatok leírásánál használatos az $\alpha|\beta|\gamma$ jelölési rendszer. Itt α a feldolgozási környezetet jelöli, itt adjuk meg a gépek típusát és számát, pl. $PD2$ azt fejezi ki, hogy 2 db párhuzamos, dedikált gép van. β jelöli a további feltételeket, γ pedig a célfüggvényt adja meg.

2.1. Ütközési feltételek

Ütközési feltételek esetén adott egy diszjunkt művelet-halmazokból álló $\mathcal{U}$ rendszer. Ha $U \in \mathcal{U}$, akkor az U-beli műveleteket ütköző műveleteknek hívjuk. Ezeket akkor sem lehet egy időben elvégezni, ha különböző gépeken vannak, különben „ütközés" jönne létre. Formálisan: $\forall U \in \mathcal{U}\ \forall \omega_1, \omega_2 \in U : C(\omega_1) \leq S(\omega_2)$ vagy $C(\omega_2) \leq S(\omega_1)$. Ha $\mathcal{U}$ nem fedi az összes műveletet, akkor a fedetlen műveleteket egyelemű halmazként $\mathcal{U}$-hoz véve ugyanazokat a feltételeket kapjuk. Tehát azt is megkövetelhetjük, hogy $\mathcal{U}$ a műveletek egy partíciója legyen.

Az ütközési feltételeket tartalmazó problémák feltételei közé további korlátokat is felvehetünk. Ezek közül kiemelem azokat, amelyekkel a későbbiekben foglalkozni fogunk.

Dedikált gépes feladatok esetén *machine chains* fogja jelölni, ha $n = m$, azaz a munkák és gépek száma megegyezik, és az i-edik munka műveleteit az i-edik gépen kell elvégezni a megadott sorrendet betartva.

A feltételek között megadhatjuk $\mathcal{U}$ maximális méretét. Ebbe azt is beleértjük, hogy $\mathcal{U}$ a műveletek egy partíciója, tehát ekkor az 1 elemű halmazokat is figyelembe kell venni.

Ha $n = m$, és az i-edik munka műveleteit az i-edik gépen végezzük, akkor $\mathcal{U}_{i,i+1}$ jelöli azt a feltételt, hogy $\mathcal{U} = \{U_{1,2}, U_{2,3}, \ldots, U_{m-1,m}\}$, ahol $U_{i,i+1} \subseteq J_i \cup J_{i+1}$, és $U_{i,i+1} = \emptyset$ is megengedett. Ilyen feltételek mellett tehát csak szomszédos gépek műveletei között lehetnek ütközési korlátok.

A bevezetésben láttuk, hogy többfejes épületnyomtatók esetén ütközőzónák kialakításával és várakozási idők beszúrásával hogyan kerülhetjük el a fejek közti ütközést. Meghatároztuk, hogy a nyomtatófej mikor lép át az ütköző zóna határán, így a nyomtatás összidejét ütköző és nem ütköző szakaszokra tudtuk bontani. A várakozások meghatározásának feladata megegyezik a $PD|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ ütemezési feladattal. Minden nyomtatófejhez tartozzon egy gép. A gép műveletei a fejhez tartozó ütköző és nem ütköző szakaszokból áll. A feldolgozási idő megegyezik ezek hosszával. Az $U_{i,i+1}$ halmaz pedig azokból a műveletekből áll, amelyek az i-edik és $i+1$-edik gép közti ütközőzónába eső ütköző szakaszoknak felelnek meg.

Párhuzamos, dedikált gépek esetén az ütközésnél általánosabb, konfliktus feltételeket is definiálhatunk. Ekkor bármely két, különböző géphez tartozó műveletre megköveteljük, hogy ne lehessen őket egy időben végezni. Ezt *conflict*-tal jelöljük. Két gép esetén a konfliktusokat megadhatjuk egy páros gráffal. Ebben az egyik gép műveletei lesznek az egyik pontosztály csúcsai, a másik gép műveletei pedig a másik pontosztály csúcsai. Két csúcs között pedig akkor húzunk be élt, ha a műveletek között konfliktus feltétel van. [6]

2.2. Job shop

Az ütemezéselmélet egyik jól ismert feladata a job shop. Az ütközéses ütemezési feladatok szorosan kapcsolódnak ehhez a problémához, emiatt ezt is megvizsgáljuk. Az $\alpha|\beta|\gamma$ jelölési rendszer esetén a job shopot az α feldolgozási környezeten belül J-vel szokás jelölni.

2.2.1. Definíció ($J||C_{max}$). Adott n munka, m gép. Az i-edik munka n_i db műveletre bomlik, ezek halmazát jelölje $J_i = \{(i,1), (i,2), \ldots, (i, n_i)\}$. Minden munkán belül a műveleteket a megadott sorrendben kell ütemezni, azaz az (i, j) művelet csak akkor kezdhető el, ha az $(i,1), (i,2), \ldots, (i, j-1)$ műveletek már elkészültek. Az (i, j) műveletet $p(i, j)$ idő alatt kell elvégezni a $\mu(i, j)$ gépen. A cél a $C_{max} = \max\{C(i, n_i) : i = 1, \ldots, n\}$ minimalizálása.

2.2.2. Tétel. *Az alábbi feladat-párok ekvivalensek egymással:*

- $J||C_{max} \Leftrightarrow PD|machine\ chains, \mathcal{U}|C_{max}$
- $Jm||C_{max} \Leftrightarrow PD|machine\ chains, |\mathcal{U}| \leq m|C_{max}$
- $J|n = k|C_{max} \Leftrightarrow PDk|machine\ chains, \mathcal{U}|C_{max}$
- $Jm|n = k|C_{max} \Leftrightarrow PDk|machine\ chains, |\mathcal{U}| \leq m|C_{max}$

Bizonyítás. Ha a job shop feladatban m gép és n munka van, akkor az ütközéses feladatban legyen n gép és n munka, $\mathcal{U} = \{U_1, U_2, \ldots, U_m\}$, $\omega \in U_i \Leftrightarrow \mu(\omega) = i$. A két feladat megoldásai között bijekció van, mert minden művelethez használhatjuk ugyanazokat a befejezési időket. Ez mindkét esetben egyértelműen meghatározza a megoldást, mert a feltételek alapján előre adott, hogy melyik művelet melyik gépre kerül. □

3. fejezet

Bonyolultság

3.1. Polinomiális algoritmus 2 gépre

Ebben a szakaszban a célunk, hogy polinomiális futási idejű algoritmust adjunk a 2 gépes $PD2|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatra. Ebben adott két gép, és $n_1 + n_2$ művelet. Az $(1,1), (1,2), \ldots, (1, n_1)$ műveleteket az első gépre kell ütemezni ebben a sorrendben, a $(2,1), (2,2), \ldots, (2, n_2)$ műveleteket pedig a második gépre szintén a megadott sorrendben. Ezen kívül adott az ütköző műveletek $U = U_{1,2}$ halmaza. A cél a C_{max} teljes átfutási idő minimalizálása.

3.1.1. Megjegyzés. A $PD2|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat speciális esete a $PD2|machine\ chains, \mathcal{U}|C_{max}$ feladatnak, ugyanis az utóbbiban $\mathcal{U}$ tetszőleges ütköző művelet-halmazokból állhat, de $\mathcal{U}_{i,i+1}$ esetén $\mathcal{U}$-ban egy $U_{1,2}$ halmazon kívül csak lényegtelen, 1-elemű halmazok lehetnek.

3.1.2. Megjegyzés. A $PD2|machine\ chains, \mathcal{U}|C_{max}$ feladat speciális esete a $PD2|machine\ chains, conflict|C_{max}$ feladatnak. Konfliktus feltételek esetén egymástól függetlenül bármely két művelet között lehet konfliktus, de az ütköző művelethalmazokban teljesülnie kell a tranzitivitásnak. Pl. ha konfliktus van a $((1,i),(2,j))$, $((2,j),(1,k))$, $((1,k),(2,l))$ művelet-párok között, akkor is lehetséges, hogy $(1,i)$ és $(2,l)$ egy időben legyenek. De ütközés esetén ez a négy művelet ugyanabban az ütköző művelet-halmazban van, és ekkor $(1,i)$ és $(2,l)$ sem lehetnek egy időben.

Az alábbiakban valójában a $PD2|machine\ chains, conflict|C_{max}$ feladatra fogunk két algoritmust adni. Az előző két megjegyzés alapján ezek használhatók a $PD2|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat megoldására is.

3.1.1. Dinamikus programozás

Először egy dinamikus programozási algoritmust adunk a feladat megoldására. Az 1. algoritmus lényegében megegyezik azzal az algoritmussal, amelyet Hong és Lin adtak a problémára [6].

3.1.3. Definíció. Egy S megengedett ütemtervet nevezzünk aktívnak, ha $\forall\omega$ műveletre és $\forall S'$ ütemtervre, amelyre $C^{S'}(\omega) < C^S(\omega)$ és $C^{S'}(\omega') = C^S(\omega')$ $\forall\omega' \neq \omega$, az S' nem megengedett.

Ha a célfüggvény a C_{max}, akkor tetszőleges megengedett megoldást aktívvá tehetünk. Vegyük a műveleteket a befejezési idők szerint növekvő sorrendben, és a következő műveletet mindig a lehető legkorábbi időpontba ütemezzünk, ahova lehet. Ezzel egyik művelet befejezési ideje sem nőhet. Ezért, ha az eredeti ütemterv optimális volt, akkor az így kapott ütemterv is az marad.

Térjünk rá a 2 gépes ütemezési feladat megoldására. Jelöljük $F[i_1, \ldots, j_1; i_2, \ldots, j_2]$-vel azt a részfeladatot, amelyet úgy kapunk, hogy az első gépen csak az $i_1, \ldots, j_1$ sorszámú műveleteket tekintjük, a második gépen az $i_2, \ldots, j_2$ sorszámúakat. Ha $i_1 > j_1$ vagy $i_2 > j_2$, akkor a megfelelő géphez egy művelet sem tartozik. A műveletek sorrendjén és a konfliktus feltételeken nem változtatunk.

Azt mondjuk, hogy az $F[i_1, \ldots, j_1; i_2, \ldots, j_2]$ feladatban nincs ütközés, ha minden műveletet szünet nélkül, közvetlenül az őt megelőző művelet után ütemezve megengedett ütemtervet kapunk. Ekkor $C^*_{max} = \max\{\sum_{i=i_1}^{j_1} p(1,i), \sum_{i=i_2}^{j_2} p(2,i)\}$.

Algoritmus 1:

$f(c_1, c_2) = \begin{cases} 0 & \text{ha } (c_1, c_2) = (0{,}0) \\ \infty & \text{különben} \end{cases}$

for $c_1 = 0, \ldots, n_1$**:**

  for $c_2 = 0, \ldots, n_2$**:**

    $f(c_1, c_2) := \min\{f(b_1, b_2) + \max\left\{\sum_{i=b_1+1}^{c_1} p(1,i), \sum_{i=b_2+1}^{c_2} p(2,i)\right\} :$
    $(0{,}0) \leq (b_1, b_2) \lneq (c_1, c_2), F[b_1+1, \ldots, c_1; b_2+1, \ldots, c_2]$-ben nincs ütközés$\}$

return $f(n_1, n_2)$

3.1.4. Tétel. *Az 1. algoritmusban $f(c_1, c_2)$ megegyezik az $F[1, \ldots, c_1; 1, \ldots, c_2]$ részfeladat optimumával. Tehát az eredeti feladatban $C^*_{max} = f(n_1, n_2)$.*

Bizonyítás. Teljes indukcióval bizonyítunk. $(c_1, c_2) = (0{,}0)$-ra teljesül az állítás. Tegyük fel, hogy minden (b_1, b_2)-re is teljesül, amelyre $(0{,}0) \leq (b_1, b_2) \lneq (c_1, c_2)$.

Ha $F[1, \ldots, c_1; 1, \ldots, c_2]$-ben nincs ütközés, akkor $f(c_1, c_2) = \max\{\sum_{i=1}^{c_1} p(1,i), \sum_{i=1}^{c_2} p(2,i)\}$. Ennél jobb megoldást biztosan nem lehet elérni, és $(b_1, b_2) = (0{,}0)$-ra megkapjuk a minimumot.

Ha $F[1, \ldots, c_1; 1, \ldots, c_2]$-ben van ütközés, akkor minden optimális ütemtervben szükség van várakozásra egy ütköző művelet előtt valamelyik gépen. Tekintsünk egy aktív optimális megoldást. Legyen $t < C_{max}$ a legutolsó időpont, amikor egy várakozás véget ér. Tegyük fel, hogy ez az első gépen lévő várakozás és $(1, i)$ az a művelet, amelyre $S(1, i) = t$.

Ekkor a második gépen biztosan van egy $(2, j)$ ütköző művelet, amelyre $C(2, j) = t$. Különben $\exists \varepsilon > 0$, hogy az $(1, i)$ szakaszt ε-nal korábban is kezdhettük volna: $S'(1, i) = S(1, i) - \varepsilon$. Ez ellentmond annak, hogy az ütemterv aktív volt.

Legyen $b_1 = i - 1, b_2 = j$. A t előtt ütemezett szakaszok megegyeznek az $F[1, \ldots, b_1; 1, \ldots, b_2]$-beli szakaszokkal, és $t = f(b_1, b_2)$, különben a t előtti ütemezést lecserélhetnénk az $f(b_1, b_2)$-t adó ütemezésre, és az optimálisnál jobb megoldást kapnánk. A t után már nincs várakozás, tehát $F[b_1 + 1, \ldots, c_1; b_2 + 1, \ldots, c_2]$-ben nincs ütközés. Így az ezekhez szükséges összidő $\max\{\sum_{i=b_1+1}^{c_1} p(1,i), \sum_{i=b_2+1}^{c_2} p(2,i)\}$. □

A minimum kiválasztásnál $O(n_1 n_2)$ esetet kell megvizsgálni, tehát a c_1, c_2-re vonatkozó ciklusokkal együtt az algoritmus futásideje $O(n_1^2 n_2^2)$.

A különbség a fenti 1. algoritmus és Hong és Lin algoritmusa [6] között, hogy ők „visszafelé” haladtak, ugyanis nem olyan részfeladatokat néztek, amelyben az első néhány művelet szerepel, hanem az olyanokat, ahol az utolsó néhány. Ezen kívül még egy észrevételt tettek, hogy a minimum kiválasztás során ne kelljen az összes (b_1, b_2)-t figyelembe venni.

Hong és Lin továbbá azt is bizonyították, hogy ha a két gép közül csak az egyik műveleteinek adott előre a sorrendje, akkor NP-nehéz problémát kapunk.

3.1.2. Geometriai megoldás

A 2 munkás $J|n = 2|C_{max}$ job shop problémára ismert egy geometriai megközelítést használó megoldás [5, 7, 8]. Ez az algoritmus valójában megoldja - a 2.2.2. tétel és a 3.1.2. megjegyzés alapján - általánosabb $PD2|machine\ chains, conflict|C_{max}$ feladatot is.

A kétdimenziós derékszögű koordináta-rendszerben tekintsük azt a T téglalapot, amelynek oldalai a tengelyekkel párhuzamosak és két szemközti csúcsa az $O = (0,0)$ és $Z = (\sum_{i=1}^{n_1} p(1,i), \sum_{i=1}^{n_2} p(2,i))$.

T-nek az x tengelyre eső oldalára mérjük fel sorban a $p(1,1), p(1,2), \ldots, p(1,n_1)$ megmunkálási időket, az y tengelyre eső oldalra pedig a $p(2,1), \ldots, p(2,n_2)$ értékeket. Jelölje $I_{1,j}$ az 1. gép j-edik munkájához tartozó intervallumot, azaz $(\sum_{i=1}^{j-1} p(1,i), \sum_{i=1}^{j} p(1,i))$-t az x tengelyen. És hasonlóan legyen $I_{2,j}$ a $(\sum_{i=1}^{j-1} p(2,i), \sum_{i=1}^{j} p(2,i))$ intervallum az y tengelyen. Ha az $(1,j)$ és $(2,k)$ műveletek között konfliktus van, akkor az $I_{1,j} \times I_{2,k}$ téglalapot tiltott téglalapnak hívjuk. A 3.1. ábrán láthatunk egy példát.

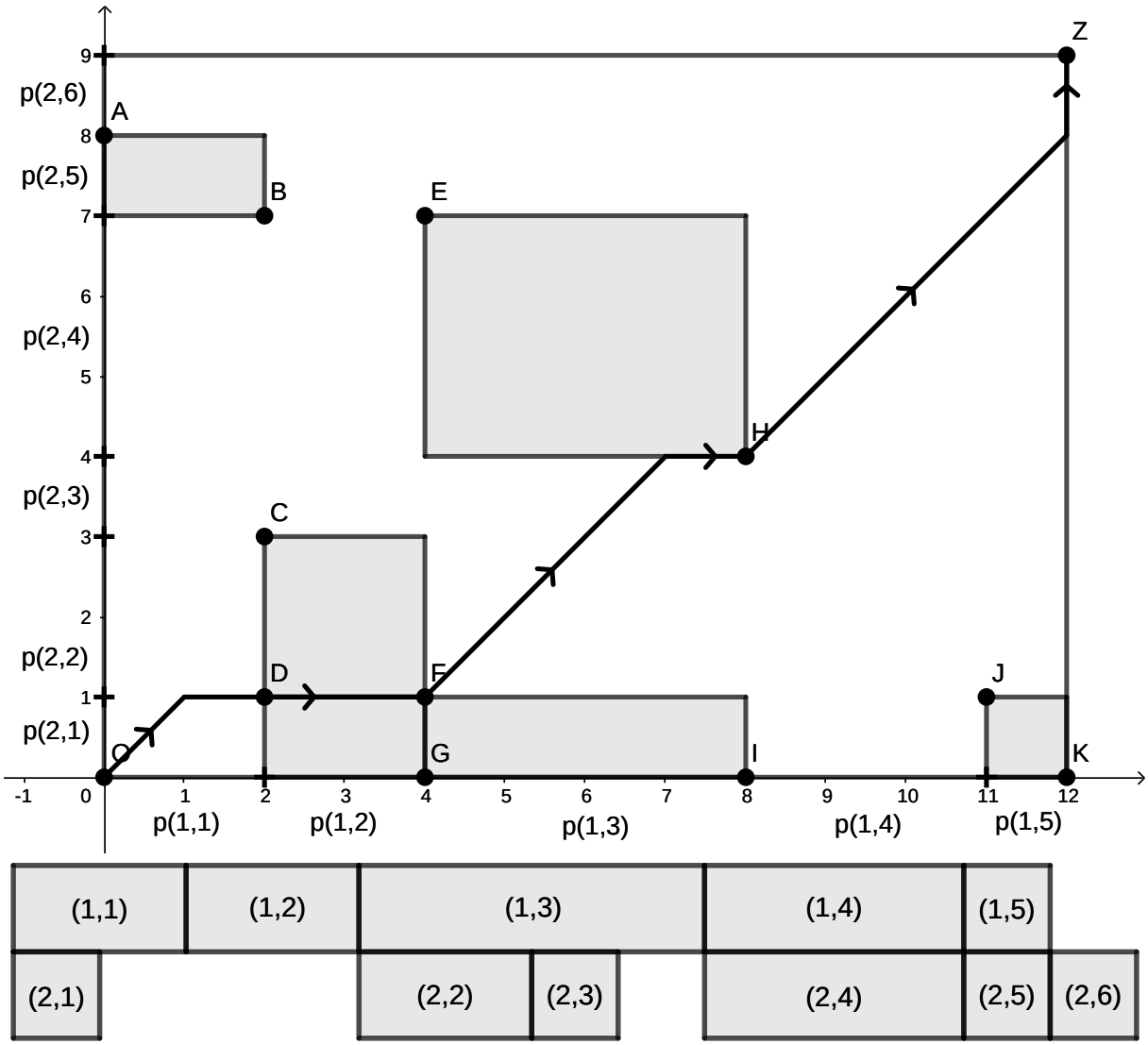
Az ütemezési feladat minden megengedett megoldásához egyértelműen tartozik egy olyan töröttvonal O-ból Z-be, amely végig T-ben halad, olyan szakaszokból áll össze, amelyek csak vízszintesek, függőlegesek, vagy átlósak (45° irányszögű) lehetnek, és nem metszi egyik tiltott téglalap belsejét sem. Ezen kívül a függőleges szakaszok csak az $I_{1,j}$ intervallumok végpontjai felett lehetnek, vízszintes szakaszok pedig csak az $I_{2,j}$ intervallumok végpontjaival egyvonalban. Ha egy töröttvonal teljesíti ezeket a feltételeket, akkor nevezzük lépcsőnek. Amikor az ütemtervben csak az első gépen dolgozunk, akkor vízszintes, amikor csak a másodikon, akkor függőleges, amikor pedig mindkét gépen egyszerre dolgozunk, akkor átlós szakaszt kapunk.

Egy lépcső hosszát úgy definiálhatjuk, hogy összeadjuk a vízszintes szakaszok összhosszát, a függőleges szakaszok összhosszát, és az átlós szakaszok összhosszának $\frac{1}{\sqrt{2}}$-szeresét. Ez megegyezik az ütemterv teljes átfutási idejével. A lépcső hosszának definíciójából látszik, hogy az optimális lépcsőben amikor csak lehet, átlósan kell haladni, és csak akkor megyünk tovább vízszintesen vagy függőlegesen, ha egy tiltott téglalap miatt nem mehetünk átlósan, vagy már elértük a T határát.

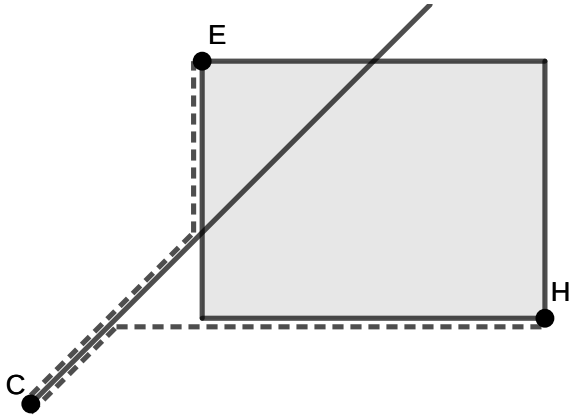
Az optimális lépcső meghatározásához készítsünk egy $D = (V, A)$ aciklikus irányított gráfot. Ennek csúcsai legyenek O, Z, és minden tiltott téglalap bal felső és jobb alsó csúcsa. Az (a, b) koordinátájú pontból kilépő éleket a következőképpen határozhatjuk meg: indítsunk egy 45° irányszögű átlós félegyenest (a, b)-ből, és legyen R az első tiltott téglalap, amibe a félegyenes belemetsz. Ekkor vegyünk fel két élt (a, b)-ből az R bal felső és jobb alsó csúcsába, ld. 3.2. ábra. Ha a félegyenes nem metsz egy tiltott téglalapot sem, mielőtt elhagyja a T-t, akkor az (a, b)-ből kivezető egyetlen él menjen Z-be. Az $(a, b) \to (c, d)$ él költsége legyen a két pont közti optimális töröttvonal hossza, azaz $\max\{c - a, d - b\}$.

D-ben egy legolcsóbb O-Z út megfelel az ütemezési feladat optimális megoldásának.

Megfelelő adatszerkezet használatával a gráf $O(r^2 \log r)$ időben megkonstruálható, ahol $r = \max\{n_1, n_2\}$. D-nek $O(r^2)$ csúcsa és $O(r^2)$ éle van, és D egy aciklikus irányított



3.1. ábra. A feladat geometriai reprezentációja. Az optimális lépcső és a hozzá tartozó ütemterv.



3.2. ábra. Az irányított gráfban a CE és CH élek a szaggatott vonallal jelölt töröttvonalakat reprezentálják.

gráf. Ezért ebben dinamikus programozás segítségével $O(r^2)$ időben meghatározható a legolcsóbb O-Z út. Így az algoritmus futásideje $O(r^2 \log r)$. [7, 8]

Ha a fenti algoritmust megfelelően módosítjuk, és megengedjük, hogy a lépcső függőleges és vízszintes szakaszai bárhol elhelyezkedjenek, akár a tiltott téglalapok belsejében is, akkor a problémának azt a változatát is meg tudjuk oldani, amikor a műveleteket meg lehet szakítani [8].

3.2. NP-nehézség 3 gépre

Az előző szakaszban látott algoritmus nem általánosítható 2-nél több gép esetére, mert ekkor általában az optimális ütemtervben nincs jó „elvágó időpont", ami mentén a feladatot két részfeladatra tudnánk bontani. Az alábbiakban a páros-páratlan partíció feladatot felhasználva bebizonyítjuk, hogy a $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat NP-nehéz.

3.2.1. Definíció (Páros-páratlan partíció)**.** Adott $e_1, \ldots, e_{2\ell}$ pozitív egészek, $\sum_{i=1}^{2\ell} e_i = 2E$.

Kérdés: Partícionálható-e az $A = \{1, \ldots, 2\ell\}$ halmaz A_1-re és A_2-re, hogy $\sum_{i \in A_1} e_i = \sum_{i \in A_2} e_i = E$ és $\{2i-1, 2i\}$-ből pontosan az egyik kerül A_1-be $\forall i = 1, \ldots, \ell$.

3.2.2. Tétel. *A páros-páratlan partíció feladat NP-nehéz.*

Bizonyítás. Az NP-teljes partíció problémát vezetjük vissza a páros-páratlan partícióra.

Ha a partíció probléma I inputja: $a_1, \ldots, a_\ell$ pozitív egészek, melyekre $\sum_{i=1}^{\ell} a_i = 2B$, akkor legyen a páros-páratlan partíció I' inputja: $e_{2k-1} = a_k + 1, e_{2k} = 1\ \forall k = 1, \ldots, \ell$, $\sum_{i=1}^{2\ell} e_i = 2(B + \ell)$.

Ha a partíció probléma megoldható, és $H_1 \cup H_2 = \{1, \ldots, \ell\}$ egy megoldás, azaz $\sum_{i \in H_1} a_i = B$, akkor $A_1 = \{2k-1 : k \in H_1\} \cup \{2k : k \in H_2\}$, $A_2 = A \setminus A_1$ a páros-páratlan partíció megoldása, ui. $\sum_{i \in A_1} e_i = \sum_{k \in H_1} (a_k + 1) + \sum_{k \in H_2} 1 = B + \ell$.

Ha a páros-páratlan partíció megoldható I'-re, és A_1, A_2 egy megoldás: $\sum_{i \in A_1} e_i = B + \ell$, akkor $H_1 = \{k : 2k-1 \in A_1, 2k \in A_2\}$, $H_2 = \{1, \ldots, \ell\} \setminus H_1$ a partíció megoldása, ui. $\sum_{k \in H_1} a_k = \sum_{i \in A_1, i \text{ páratlan}} (e_i - 1) = \sum_{i \in A_1} (e_i - 1) = B + \ell - |A_1| = B$. □

3.2.3. Tétel. *A $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat NP-nehéz.*

Sotskov és Shakhlevich [9]-ben bebizonyították, hogy a $J3|n = 3|C_{max}$ job shop probléma NP-nehéz. A következő bizonyítás alapötlete innen származik.

Bizonyítás. A páros-páratlan partíciót visszavezetjük az ütemezési feladatra.

Adott $e_1, \ldots, e_{2\ell} \in \mathbb{Z}_+$, $\sum_{i=1}^{2\ell} e_i = 2E$. Feltehető, hogy $e_{2k+2} < e_{2k+1}\ \forall k = 0, \ldots, \ell - 1$, ui. $e_{2k+2} = e_{2k+1}$ esetén elhagyhatjuk, $e_{2k+2} > e_{2k+1}$ esetén pedig felcserélhetjük őket. $\Delta_{2k+1,2k+2} := e_{2k+1} - e_{2k+2} > 0\ \forall k = 0, \ldots, \ell - 1$. Legyen $H \geq 3E$ konstans.

Legyen $n_1 = 4\ell, n_2 = 5\ell, n_3 = 5\ell$. A feldolgozási idők ($k = 0, \ldots, \ell - 1$):

$p(1,4k+1) = 2H + e_{2k+2}$	$p(2,5k+1) = e_{2k+1}$	$p(3,5k+1) = H$
$p(1,4k+2) = \Delta_{2k+1,2k+2}$	$p(2,5k+2) = E$	$p(3,5k+2) = e_{2k+1}$
$p(1,4k+3) = 2H - \Delta_{2k+1,2k+2}$	$p(2,5k+3) = e_{2k+2}$	$p(3,5k+3) = 2H - e_{2k+1}$
$p(1,4k+4) = 2H$	$p(2,5k+4) = \Delta_{2k+1,2k+2}$	$p(3,5k+4) = 2H$
	$p(2,5k+5) = H$	$p(3,5k+5) = H$

Az ütköző művelethalmazok pedig az alábbiak:

$$U_{1,2} = \bigcup_{k=0}^{\ell-1} \{(1,4k+1), (1,4k+3), (1,4k+4), (2,5k+4)\}$$

$$U_{2,3} = \bigcup_{k=0}^{\ell-1} \{(2,5k+1), (2,5k+3), (2,5k+5), (3,5k+1), (3,5k+3), (3,5k+4)\}$$

3.2.4. Lemma. *Ennek az ütemezési feladatnak pontosan akkor van olyan megoldása, amelyre $C_{max} \leq 6\ell H + E$, ha a páros-páratlan partíció megoldható.*

Bizonyítás. A műveletek halmazát ℓ db részhalmazra bontjuk, ezeket blokkoknak fogjuk hívni. A $k+1$-edik blokk: $B_{k+1} := \{(1,4k+1), \ldots, (1,4k+4), (2,5k+1), \ldots, (2,5k+5), (3,5k+1), \ldots, (3,5k+5)\}$ $(k = 0, \ldots, \ell-1)$.

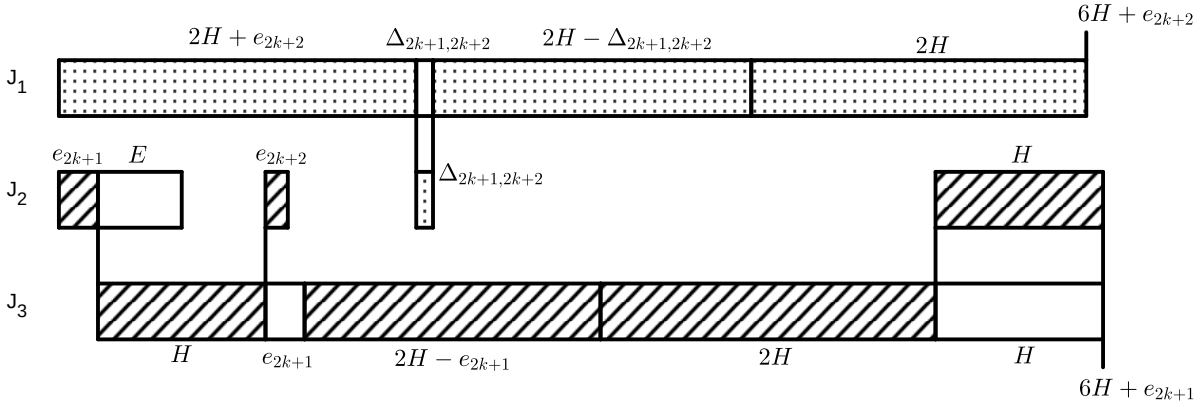
Az ω, ω' műveletekre jelölje $\omega \to \omega'$ azt, hogy $C(\omega) \leq S(\omega')$.

Elégségesség:

Az elégségesség bizonyítása során először a blokkok kétféle ütemtervét fogjuk definiálni. Ezután a páros-páratlan partíció megoldása alapján minden blokkhoz kiválasztjuk, hogy a kettő közül melyik ütemtervet használjuk, és ezeket összerakva egy olyan megengedett megoldást állítunk elő, amelyre $C_{max} \leq 6\ell H + E$.

A B_{k+1} blokk I. típusú ütemtervében az $U_{1,2}$-beli műveleteket $(1,4k+1) \to (2,5k+4) \to (1,4k+3) \to (1,4k+4)$ sorrendben ütemezzük, az $U_{2,3}$-beli műveleteket pedig $(2,5k+1) \to (3,5k+1) \to (2,5k+3) \to (3,5k+3) \to (3,5k+4) \to (2,5k+5)$ sorrendben. (ld. 3.3. ábra). Minden műveletet lokálisan kezdjünk a lehető leghamarabb, pl. $S(2,5k+3) = C(3,5k+1)$.

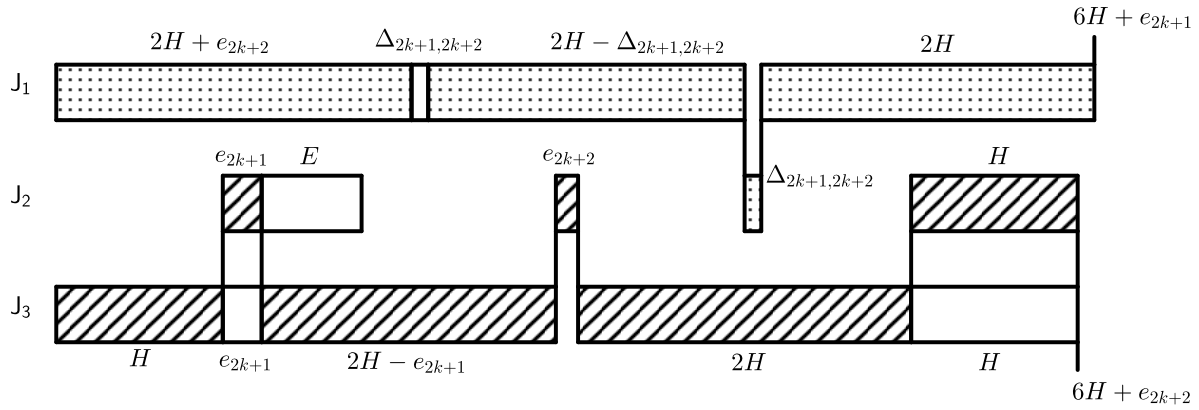
$C(1,4k+4) = S(1,4k+1) + 6H + e_{2k+2}$, és $C(2,5k+5) = C(3,5k+5) = S(2,5k+1) + 6H + e_{2k+1}$.



3.3. ábra. A B_{k+1} blokk I. típusú ütemterve

A B_{k+1} blokk II. típusú ütemtervében az $U_{1,2}$-beli műveleteket $(1,4k+1) \to (1,4k+3) \to (2,5k+4) \to (1,4k+4)$ sorrendben ütemezzük, az $U_{2,3}$-beli műveleteket pedig $(3,5k+1) \to (2,5k+1) \to (3,5k+3) \to (2,5k+3) \to (3,5k+4) \to (2,5k+5)$ sorrendben. (ld. 3.4. ábra). Itt is minden műveletet lokálisan kezdjünk a lehető leghamarabb.

$C(1,4k+4) = S(1,4k+1) + 6H + e_{2k+1}$, és $C(2,5k+5) = C(3,5k+5) = S(3,5k+1) + 6H + e_{2k+2}$.



3.4. ábra. A B_{k+1} blokk II. típusú ütemterve

A 3.3. és 3.4. ábrákon a gépek közti függőleges vonalakkal jelöltem az olyan időpontokat, amelyek garantáltan egybeesnek a két gépen. Vegyük észre, hogy az ütemtervek elején nincs ilyen jelölés, ugyanis ezek általában nem fognak egyszerre kezdődni.

Legyen $A_1 \cup A_2$ a páros-páratlan partíció megoldása. Ha $e_{2k+1} \in A_1, e_{2k+2} \in A_2$, akkor használjuk a B_{k+1} blokk I. típusú ütemtervét, különben a II. típusút. A blokkokat közvetlenül egymás után ütemezzük, azaz $S(1{,}4k+1) = C(1{,}4k)$ és $\min\{S(2{,}5k+1), S(3{,}5k+1)\} = C(2{,}5k) = C(3{,}5k)\ \forall k$.

Így egy megengedett ütemtervet kapunk, ugyanis az $U_{1,2}$, ill. $U_{2,3}$-beli műveletek nem lógnak egymásba. Az 1. és 3. munka műveletei garantáltan jó sorrendben vannak. És a 2. munka esetén is csak azt kell ellenőriznünk, hogy $C(2{,}5k+3) \leq S(2{,}5k+4)$, $C(2{,}5k+4) \leq S(2{,}5k+5)\ \forall k$. Ehhez a következő egyenlőtlenségeket használjuk fel:

$$C(1{,}4k) = 6kH + \sum_{\substack{i=1\\ i\in A_2}}^{2k} e_i \begin{cases} \geq 6kH + \sum_{j=0}^{k-1} e_{2j+2} \\ \leq 6kH + \sum_{j=0}^{k-1} e_{2j+1} \end{cases} \quad \forall k$$

$$C(2{,}5k) = C(3{,}5k) = 6kH + \sum_{\substack{i=1\\ i\in A_1}}^{2k} e_i \begin{cases} \geq 6kH + \sum_{j=0}^{k-1} e_{2j+2} \\ \leq 6kH + \sum_{j=0}^{k-1} e_{2j+1} \end{cases} \quad \forall k$$

Ha B_{k+1} I. típusú:

$$C(2,5k+3) = C(2{,}5k) + p(2{,}5k+1) + p(3{,}5k+1) + p(2{,}5k+3) =$$
$$= C(2{,}5k) + e_{2k+1} + H + e_{2k+2} \leq 6kH + \sum_{j=0}^{k} e_{2j+1} + H + e_{2k+2}$$

$$S(2{,}5k+4) = C(1{,}4k) + p(1{,}4k+1) = C(1{,}4k) + 2H + e_{2k+2} \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + 2H$$

$$\Rightarrow S(2{,}5k+4) - C(2{,}5k+3) \geq H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} - e_{2k+2} \geq 0$$

$$C(2{,}5k+4) = S(2{,}5k+4) + p(2{,}5k+4) = C(1{,}4k) + 2H + e_{2k+1} \leq 6kH + \sum_{j=0}^{k} e_{2j+1} + 2H$$

$$S(2,5k+5) = C(2,5k)+p(2,5k+1)+p(3,5k+1)+p(3,5k+2)+p(3,5k+3)+p(3,5k+4) =$$

$$= C(2,5k) + 5H + e_{2k+1} \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + 5H$$

$$\Rightarrow S(2,5k+5) - C(2,5k+4) \geq 3H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} \geq 0$$

Ha B_{k+1} II. típusú:

$$C(2,5k+3) = C(3,5k) + p(3,5k+1) + p(3,5k+2) + p(3,5k+3) + p(2,5k+3) =$$

$$= C(3,5k) + 3H + e_{2k+2} \leq 6kH + \sum_{j=0}^{k} e_{2j+1} + 3H$$

$$S(2,5k+4) = C(1,4k) + p(1,4k+1) + p(1,4k+2) + p(1,4k+3) =$$

$$= C(1,4k) + 4H + e_{2k+2} \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + 4H$$

$$\Rightarrow S(2,5k+4) - C(2,5k+3) \geq H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} \geq 0$$

$$C(2,5k+4) = S(2,5k+4)+p(2,5k+4) = C(2,5k+4)+4H+e_{2k+1} \leq 6kH+\sum_{j=0}^{k} e_{2j+1}+4H$$

$$S(2,5k+5) = C(3,5k)+p(3,5k+1)+p(3,5k+2)+p(3,5k+3)+p(2,5k+3)+p(3,5k+4) =$$

$$= C(3,5k) + 5H + e_{2k+2} \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + 5H$$

$$\Rightarrow S(2,5k+5) - C(2,5k+4) \geq H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} \geq 0$$

Tehát a kapott ütemterv megengedett. És $C_{max} \leq 6\ell H + E$ is teljesül:

$$C(1,4\ell) = 6\ell H + \sum_{i \in A_2} e_i = 6\ell H + E$$

$$C(2,5\ell) = C(3,5\ell) = 6\ell H + \sum_{i \in A_1} e_i = 6\ell H + E$$

Szükségesség:

Először be fogjuk látni, hogy ha egy ütemtervre $C_{max} \leq 6\ell H + E$, akkor az $U_{1,2} \cup U_{2,3}$-beli műveletek ütemezése felbontható a blokkok szerint, azaz ha $k < k'$, $\omega \in B_k \cap U_{1,2}$,

$\omega' \in B_{k'} \cap U_{1,2}$ vagy $\omega \in B_k \cap U_{2,3}$, $\omega' \in B_{k'} \cap U_{2,3}$, akkor $\omega \to \omega'$. Ezután elég a blokkok ütemtervét külön-külön meghatározni. Megmutatjuk, hogy a $C_{max} \leq 6\ell H + E$-t teljesítő ütemterv választható úgy, hogy minden blokk a korábban definiált I. vagy II. típusú ütemterv szerint legyen ütemezve. Ebből végül megkapjuk a páros-páratlan partíció megoldását.

$$\sum_{i=1}^{4\ell} p(1,i) = 6\ell H + \sum_{k=0}^{\ell-1} e_{2k+2}, \text{ és } \sum_{i=1}^{5\ell} p(3,i) = 6\ell H$$

Tehát egy olyan ütemtervben, ahol $C_{max} \leq 6\ell H + E$, az 1. gépen összesen legfeljebb $\delta := E - \sum_{k=0}^{\ell-1} e_{2k+2} = \frac{1}{2}\sum_{k=0}^{\ell-1} \Delta_{2k+1,2k+2}$ ideig várakozhatunk, a 3. gépen pedig legfeljebb E ideig.

A következő állítás az $M := \{(2,5k+5) : k = 0, \ldots, \ell-1\}$-beli műveletek lehetséges elhelyezkedéséről szól.

3.2.5. Állítás. *$(3,5k+4) \to (2,5k+5) \to (3,5k+6)$ $\forall k = 0, \ldots, \ell-2$, és $(3,5\ell - 1) \to (2,5\ell)$.*

Bizonyítás. A skatulya-elvet fogjuk használni. Először belátjuk, hogy a $(2,5k+5) \in M$ műveletet csak a következő helyekre ütemezhetjük: $(3,5j+4) \to (2,5k+5) \to (3,5j+6)$ valamelyik j-re, vagy $(3,5\ell-1) \to (2,5k+5)$.

Ha indirekt $(2,5) \to (3,1)$, akkor a 3. gépen legalább $H + E + 2e_1 > E$ várakozásra lenne szükség $(3,1)$ előtt.

Legyen $(3,j), (3,j') \in U_{2,3}$ két művelet a 3. munkában, melyekre $(j,j') \in \{(5k+1, 5k+3) : k = 0, \ldots, \ell-1\} \cup \{(5k+3, 5k+4) : k = 0, \ldots, \ell-1\}$. És legyen $(2,i) \in M$. Ha indirekt $(3,j) \to (2,i) \to (3,j')$, akkor a 3. gépen legalább $H - \max\{e_{2k+1} : k = 0, \ldots, \ell-1\} > E$ várakozásra lenne szükség $(3,j)$ és $(3,j')$ között.

Tehát az M-beli műveletek valóban csak a fent definiált „skatulyákba" kerülhetnek. Az alábbiakban belátjuk, hogy mindegyik „skatulyába" legfeljebb 1 műveletet ütemezhetük az M halmazból, így megkapjuk az állítást.

Ha indirekt $(3,5j+4) \to (2,5k) \to (2,5k+5) \to (3,5j+6)$, akkor a 3. gépen legalább $2e_{2k+1} + E + H > H$ várakozásra lenne szükség $(3,5j+4)$ és $(3,5j+6)$ között.

Ha indirekt $(3,5\ell-1) \to (2,5\ell-5)) \to (2,5\ell)$, akkor $C(2,5\ell) \geq C(3,5\ell-1) + p(2,5\ell-5) + p(2,5\ell-4) + \cdots + p(2,5\ell) \geq (6\ell-1)H + 2H + E + 2e_{2\ell-1} > 6\ell H + E$. □

3.2.6. Következmény. *Ha $k < k'$, $\omega \in B_k \cap U_{2,3}$, $\omega' \in B_{k'} \cap U_{2,3}$, akkor $\omega \to \omega'$.*

Az alábbiakban vizsgáljuk meg az $U_{1,2}$-beli ütköző műveletek lehetséges elhelyezkedését is.

3.2.7. Állítás. $(1,4k+1) \to (2,5k+4) \to (1,4k+4)$ $\forall k = 0, \ldots, \ell-1$.

Bizonyítás. Ha indirekt $(2,5k+4) \to (1,4k+1)$, akkor az 1. gépen δ-nál több várakozás lenne:

$$S(1,4k+1) \geq C(2,5k) + p(2,5k+1) + p(2,5k+2) + p(2,5k+3) + p(2,5k+4) \geq$$
$$\geq \sum_{j=1}^{k} \sum_{\omega \in B_j \cap U_{2,3}} p(\omega) + E + 2e_{2k+1} \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + E$$

$$S(1,4k+1) - \sum_{i=1}^{4k} p(1,i) \geq 6kH + \sum_{j=0}^{k} e_{2j+2} + E - 6kH - \sum_{j=0}^{k-1} e_{2j+2} = E + e_{2k+2} > \delta$$

Ha indirekt $(1{,}4k+4) \to (2{,}5k+4)$:

$$C(2{,}5k{+}5) \geq \sum_{i=1}^{4k+4} p(1,i) + p(2{,}5k{+}4) + p(2{,}5k{+}5) = 6(k{+}1)H + \sum_{j=0}^{k} e_{2j+2} + \Delta_{2k+1,2k+2} + H$$

Ha $k < \ell - 1$, akkor a 3. gépen E-nél több várakozás lenne:

$$S(3{,}5k+6) - \sum_{i=1}^{5k+5} p(3,i) \geq C(2{,}5k+5) - 6(k+1)H \geq \sum_{j=0}^{k} e_{2j+2} + \Delta_{2k+1,2k+2} + H > E$$

Ha $k = \ell - 1$, akkor $C(2{,}5\ell) \geq 6\ell H + \sum_{j=0}^{\ell-1} e_{2j+2} + \Delta_{2\ell-1,2\ell} + H > 6\ell H + E$ lenne. $\square$

3.2.8. Következmény. *Ha $k < k'$, $\omega \in B_k \cap U_{1,2}$, $\omega' \in B_{k'} \cap U_{1,2}$, akkor $\omega \to \omega'$.*

A 3.2.6. és 3.2.8. következmények alapján az ütköző műveleteket elég blokkonként tekinteni. Nézzük a B_{k+1} blokkot.

Az $U_{1,2}$-beli műveletekre az $(1{,}4k+1) \to (2{,}5k+4) \to (1{,}4k+3)$ sorrendet hívjuk I.1. sorrendnek, a $(1{,}4k+3) \to (2{,}5k+4) \to (1{,}4k+4)$ sorrend pedig II.1. sorrendnek. Ezek szerepelnek a B_{k+1} I., ill. II. típusú ütemtervében. A 3.2.7. állítás miatt más lehetőség nincs.

Ha a célfüggvény C_{max}, akkor mindig van olyan optimális ütemterv, amely aktív (ld. 3.1.3. definíció), ezért a $C_{max} \leq 6\ell H + E$-t teljesítő ütemtervet választhatjuk az aktív ütemtervek közül.

Az $U_{2,3}$-beli műveletek lehetséges sorrendje alapján 5 esetet különböztethetünk meg.

1. $(3,i) \to (2{,}5k+1) \to (2{,}5k+3) \to (3,j)$, ahol $(i,j) \in \{(5k{,}5k+1), (5k+1{,}5k+3), (5k+3{,}5k+4), (5k+4{,}5k+6)\}$.

   Ekkor a B_{k+1} blokkban a 3. gépen legalább $E + e_{2k+2} > E$ várakozás van, tehát ez nem lehet.

2. $(3{,}5k+4) \to (2{,}5k+3)$.

   Felhasználjuk, hogy a 3.2.7. állítás alapján $(2{,}5k+4) \to (1{,}4k+4)$.

   $$C(1{,}4k{+}4) \geq \sum_{i=1}^{5k+4} p(3,i) + p(2{,}5k{+}3) + p(2{,}5k{+}4) + p(1{,}4k{+}4) = (6k{+}7)H + e_{2k+1}$$

   $$C(1{,}4k{+}4) - \sum_{i=1}^{4k+4} p(1,i) \geq (6k{+}7)H + e_{2k+1} - 6(k{+}1)H - \sum_{j=0}^{k} e_{2j+2} \geq H - \sum_{i=0}^{k} e_{2i+2} > \delta$$

   Ekkor az 1. gépen a megengedett δ-nál több várakozás van, tehát ez nem fordulhat elő.

3. $(2{,}5k+1) \to (3{,}5k+1)$, és $(3{,}5k+3) \to (2{,}5k+3)$.

   Ekkor az ütemterv nem aktív, mert $(2{,}5k{+}2)$-t ütemezhetjük $S(3{,}5k{+}1)$-től kezdve, és a $(2{,}5k+3)$-at ütemezhetjük korábban, az $S(3{,}5k+2)$ időponttól kezdve.

4. $(2{,}5k+1) \to (3{,}5k+1) \to (2{,}5k+3) \to (3{,}5k+3)$.

   Ezt a sorrendet hívjuk I.2. sorrendnek. Ez szerepel a B_{k+1} I. típusú ütemtervében.

5. $(3{,}5k+1) \to (2{,}5k+1) \to (3{,}5k+3) \to (2{,}5k+3) \to (3{,}5k+4)$.

   Ezt hívjuk II.2 sorrendnek, ez szerepel a B_{k+1} blokk II. típusú ütemtervében.

Eddig csak az ütköző műveletek sorrendjéről volt szó, azonban aktív ütemterv esetén ezek az ütköző és nem ütköző műveletek helyét is egyértelműen megadják:

Az I.1., ill. a II.1 sorrendet használva csak úgy lesz aktív az ütemterv, ha a $(B_{k+1} \cap \cap U_{1,2}) \cup \{(1{,}4k+2)\}$ műveleteket a B_{k+1} blokk I., ill. II. típusú ütemtervének megfelelően ütemezzük.

Az I.2., ill. a II.2. sorrendet betartó aktív ütemtervben pedig a $(B_{k+1} \cap U_{2,3}) \cup \{(2{,}5k+ + 2), (3{,}5k+2), (3{,}5k+5)\}$ műveleteket az I., ill. II. típusú ütemtervnek megfelelően kell ütemezni.

Az I.1. sorrend jobb, mint a II.1., ui. a B_{k+1} blokk ütemtervében $C^{I.1}(1{,}5k+5) - - S^{I.1}(1{,}5k+1) = 6H + e_{2k+2} < 6H + e_{2k+1} = C^{II.1}(1{,}5k+5) - S^{II.1}(1{,}5k+1)$. És hasonlóan látható, hogy a II.2. sorrend jobb, mint az I.2.

Az I.1. és II.2. azonban egyszerre nem fordulhat elő:

Az I.1. esetén az 1. gépen nincs várakozás. Ezt csak úgy érhetjük el, ha

$$S(2{,}5k+4) = C(1{,}4k+1) \leq \sum_{i=1}^{4k+1} p(1,i) + \delta = 6kH + \sum_{j=0}^{k-1} e_{2j+2} + 2H + e_{2k+2} + \delta$$

II.2-ben a 3. gépen e_{2k+2} várakozás van. Ezt csak úgy érhetjük el, ha

$$C(2{,}5k+3) = S(3{,}5k+1) + p(3{,}5k+1) + p(3{,}5k+2) + p(3{,}5k+3) + p(2{,}5k+3) \geq$$
$$\geq \sum_{j=1}^{k} \sum_{\omega \in B_j \cap U_{2,3}} p(\omega) + 3H + e_{2k+2} = 6kH + \sum_{j=0}^{k-1} e_{2j+2} + 3H + e_{2k+2}$$

Ekkor viszont megsérülne a 2. gépen a műveletek sorrendjére vonatkozó feltétel: $S(2{,}5k+ + 4) - C(2{,}5k+3) \leq \delta - H < 0$.

Ha egy megoldásban a B_{k+1} blokk műveletei a II.1 és I.2 sorrendeknek megfelelően helyezkednek el, akkor a II.1-et I.1-re cserélve egy nem rosszabb megoldást kapunk.

A fentiek alapján van olyan ütemterv, hogy $C_{max} \leq 6\ell H + E$, és minden blokk az I. vagy a II. típusú ütemterv szerint van ütemezve. Ha a B_{k+1} I. típusú, akkor legyen $e_{2k+1} \in A_1$, különben $e_{2k+2} \in A_1$.

$$6\ell H + E \geq C_{max} \geq C(1{,}4\ell) = 6\ell H + \sum_{i \in A_2} e_i$$

$$6\ell H + E \geq C_{max} \geq C(3{,}5\ell) = 6\ell H + \sum_{i \in A_1} e_1$$

Tehát $\sum_{i \in A_1} e_i \leq E$, és $\sum_{i \in A_2} e_i \leq E$. Mivel $\sum_{i=1}^{2\ell} e_i = 2E$, ezért ez csak úgy lehet, ha $\sum_{i \in A_1} e_i = \sum_{i \in A_1} e_i = E$, tehát a páros-páratlan partíció egy megoldását kaptuk.

□

Ezzel a lemma bizonyítását befejeztük. Mivel a páros-páratlan partíció feladat NP-nehéz (3.2.2. tétel), így következik, hogy a $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat is NP-nehéz. □

3.3. Job shop NP-nehézsége 3 gép és 3 munka esetén

Tekintsük a job shop problémának azon változatát, ahol a gépek és a munkák száma is 3. Sotskov és Shakhlevich bizonyították, hogy ez a feladat NP-nehéz [9]. Az előző szakaszban vizsgált $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat nagyon hasonlít a job shopra. Ezért most a 3.2.3. tétel bizonyítását kiegészítve, a [9]-ben szereplő eredeti bizonyításhoz hasonló, de annál valamivel egyszerűbb, új bizonyítást adunk a következő tételre.

3.3.1. Tétel (Sotskov, Shakhlevich). *A $J3|n = 3|C_{max}$ job shop probléma NP-nehéz.*

Bizonyítás. A 2.2.2 tétel alapján a $J3|n = 3|C_{max}$ és a $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladatok ekvivalensek. Mi az utóbbiról fogjuk belátni, hogy NP-nehéz.

Legyen a páros-páratlan partíció I inputja a következő: $e_1, \ldots, e_{2\ell} \in \mathbb{Z}_+$, $\sum_{i=1}^{2\ell} e_i = 2E$, ahol $e_{2k+2} < e_{2k+1}$ $\forall k = 0, \ldots, \ell - 1$. És legyen $H \geq 3E$ konstans.

A 3.2.3. tétel bizonyításában ehhez definiáltuk a $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat I' inputját.

A $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladat I'' inputjában a műveletek száma és feldolgozási ideje legyen ugyanaz, mint I'-ben, az ütköző művelethalmazok pedig: $U_1 = U_{1,2}$, $U_2 = U_{2,3}$, $U_3 = (J_1 \cup J_2 \cup J_3) \setminus (U_{1,2} \cup U_{2,3})$.

3.3.2. Lemma. *Az alábbiak ekvivalensek:*

(a) A páros-páratlan partíciónak van megoldása az I inputra.

(b) A $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatnak az I' inputra van olyan megoldása, amelyre $C_{max} \leq 6\ell H + E$.

(c) A $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladatnak az I'' inputra van olyan megoldása, amelyre $C_{max} \leq 6\ell H + E$.

Bizonyítás. A 3.2.4. lemmában bizonyítottuk, hogy $(a) \Leftrightarrow (b)$.

$(a) \Rightarrow (c)$

A 3.2.4. lemma bizonyításához hasonlóan használjuk most is a blokkok I. és II. típusú ütemtervét (ld. 3.3. és 3.4. ábra).

Ha a páros-páratlan partíció megoldása A_1, A_2, akkor $e_{2k+1} \in A_1, e_{2k+2} \in A_2$ esetén használjuk a B_{k+1} blokk I. típusú ütemtervét, különben a II. típusút. A blokkokat közvetlenül egymás után ütemezzük. A 3.2.4. lemmában bizonyítottuk, hogy ekkor az ütemterv betartja a műveletek sorrendjére vonatkozó szabályokat, továbbá az U_1 és U_2-beli műveletek nem lógnak egymásba. Sőt, azt is tudjuk, hogy a kapott ütemtervben $C_{max} \leq 6\ell H + E$. Így itt már csak azt kell ellenőriznünk, hogy az U_3 műveletei is diszjunktak lesznek.

Ha B_{k+1} I. típusú, akkor az U_3-beli műveletek sorrendje: $(2{,}5k+2) \to (3{,}5k+2) \to (1{,}4k+2) \to (3{,}5k+5)$.

$$C(3{,}5k) = C(2{,}5k) \leq S(2{,}5k+1) < S(2{,}5k+2)$$
$$C(2{,}5k+2) \leq S(2{,}5k+3) = S(3{,}5k+2)$$

$$S(1{,}4k+2) - C(3{,}5k+2) \geq \left(6kH + \sum_{j=0}^{k} e_{2j+2} + 2H\right) - \left(6kH + \sum_{j=0}^{k} e_{2j+1} + H + e_{2k+1}\right) =$$
$$= H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} - e_{2k+1} \geq 0$$

$$C(1{,}4k+2) = C(2{,}5k+4) \leq S(2{,}5k+5) = S(3{,}5k+5)$$

Ha B_{k+1} II. típusú, akkor az U_3-beli műveletek sorrendje: $(3{,}5k+2) \to (2{,}5k+2) \to (1{,}4k+2) \to (3{,}5k+5)$.

$$C(3{,}5k) \leq S(3{,}5k+1) < S(3{,}5k+2)$$
$$C(3{,}5k+2) = S(2{,}5k+2)$$

$$S(1{,}4k+2) - C(2{,}5k+2) \geq \left(6kH + \sum_{j=0}^{k} e_{2j+2} + 2H\right) - \left(6kH + \sum_{j=0}^{k} e_{2j+1} + H + E\right) =$$
$$= H - \sum_{j=0}^{k} \Delta_{2j+1,2j+2} - E \geq 0$$

$$C(1{,}4k+2) < C(1{,}4k+3) = S(2{,}5k+4) < S(2{,}5k+5) = S(3{,}5k+5)$$

Tehát a kapott ütemterv megengedett.

$(c) \Rightarrow (b)$

Az I'-re vonatkozó $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladathoz képest a $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladatban I''-re az egyetlen plusz feltétel az, hogy az U_3-beli műveleteknek is diszjunktaknak kell lenni. A többi feltétel megegyezik a két feladatban.

Tehát ha a $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladatnak van olyan megoldása I''-re, amelyben $C_{max} \leq 6\ell H + E$, akkor ugyanez az ütemterv jó lesz $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$-ra és I'-re is. □

A lemma $(a) \Leftrightarrow (c)$ ekvivalenciája mutatja, hogy a $PD3|machine\ chains, |\mathcal{U}| \leq 3|C_{max}$ feladat NP-nehéz. □

4. fejezet

Közelítő algoritmusok

Ha $P \neq NP$, akkor NP-nehéz problémák esetén nem tudunk olyan algoritmust adni, amely polinom időben meghatározná az optimális megoldást. Ilyenkor megpróbálhatunk olyan algoritmust kidolgozni, amely az optimálishoz közeli megoldást ad.

4.0.1. Definíció. Egy minimalizálási problémára adott algoritmust α-közelítőnek vagy α-approximációnak hívunk, ha az általa adott megoldás legfeljebb az optimum α-szorosa.

A továbbiakban az alábbi jelöléseket fogjuk használni:

$$P_i := \sum_{j=1}^{n_i} p(i,j)$$

$$P_{max} := \max\{P_i : i = 1, \ldots, m\}$$

4.1. 2-approximáció

Először egy egyszerű 2-approximációs algoritmust adunk a $PD|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ problémára.

Algoritmus 2: 2-approximáció

1 **ciklus** $i = 1, \ldots, m$, *i páratlan*:
2 Az i-edik gép műveleteit ütemezzük szorosan egymás után a 0 időponttól kezdve.
3 **ciklus** $i = 2, \ldots, m$, *i páros*:
4 Az i-edik gépen először várjunk $\max\{P_{i-1}, P_{i+1}\}$ ideig, és ezután a műveleteket ütemezzük közvetlenül egymás után.

4.1.1. Lemma. *A $PD|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat optimális megoldására teljesül, hogy $P_{max} \leq C^*_{max} \leq 2P_{max}$.*

Bizonyítás. Az i-edik gépen az összes műveletet ütemezni kell, ezért minden megengedett ütemtervben $C(i, n_i) \geq P_i$.

$$C^*_{max} = \max\{C^*(i, n_i) : i = 1, \ldots, m\} \geq \max\{P_i : i = 1, \ldots, m\} = P_{max}$$

A 2. algoritmussal olyan megengedett ütemtervhez jutunk, amelyben

$$C_{max} = \max\{P_i + P_{i+1} : i = 1, \ldots, m-1\} \leq 2P_{max}$$

Tehát $C^*_{max} \leq C_{max} \leq 2P_{max}$. □

A lemma bizonyításából egyszerűen következik az alábbi állítás.

4.1.2. Állítás. *A 2. algoritmus 2-approximáció a $PD|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatra.*

4.2. 5/3-approximáció 3 gépre

Ebben a részben egy 5/3-approximációs algoritmust adunk a $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatra.

A megoldás során kihasználjuk, hogy a $PD2|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladat polinom időben megoldható. Az algoritmus során sok különböző inputra meg kell határoznunk ennek a 2 gépes feladatnak az optimumát. Ha B_1 az i-edik gép műveleteinek egy részhalmaza, B_2 az $(i+1)$-edik gép műveleteinek részhalmaza, és β egy nem ütköző művelet, akkor pl. $[B_1 \text{ vissza}, B_2 + \beta]$ jelöli azt az inputot, hogy az 1. gépre a B_1 műveleteit kell ütemeznünk az eredetihez képest fordított sorrendben, a 2. gépre pedig a B_2 műveleteit kell ütemezni az eredeti sorrendet megtartva, és ezután β-t mint utolsó műveletet. Az ütköző műveletek megegyeznek az eredeti $U_{i,i+1}$-beli ütköző műveletekkel, a többi művelet nem ütköző.

4.2.1. Tétel. *A 3. algoritmus 5/3-approximáció a $PD3|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatra.*

Bizonyítás. Először nézzük a 2. sorban szereplő esetet. Az itt szereplő ütemterv megengedett, mivel az első két gépre adott megoldásban az eltolás után sem lehet ütközés. A 2. gépen csak a $(2,k)$ feldolgozása esik az $[S(2,k), C(2,k) + (P_3 - p(2,k))]$ intervallumba, de $(2,k) \notin U_{2,3}$, ezért a 2. és 3. gép között sem lehet ütközés. Az első két gépre vonatkozó optimális megoldás teljes átfutási idejére teljesül, hogy legfeljebb C^*_{max}. A 4.1.1. lemma alapján $C^{alg}_{max} \leq C^*_{max} + P_3 - p(2,k) \leq C^*_{max} + \frac{2}{3}P_{max} \leq \frac{5}{3}C^*_{max}$.

A 6. sorban lévő eset az előzőhöz hasonló.

Végül ha a 10. sorban lévő feltétel teljesül, akkor egy konkrét 2a-2b kettéosztásra fogjuk belátni, hogy $C_{max} \leq \frac{5}{3}C^*_{max}$.

Tekintsünk egy optimális megoldást a 3 gépre. Legyen t a $\frac{1}{2}C^*_{max}$-hoz legközelebbi időpont, amire teljesül, hogy $C^*(2,j) \leq t$ vagy $S^*(2,j) \geq t$ minden $j = 1, \ldots, n_2$-re. A 2. gépen lévő műveleteket két halmazba osztjuk: a 2a blokkba kerülnek a t előtt befejeződő műveletek, és a 2b blokkba kerülnek a t után kezdődő műveletek, ld. 4.1. ábra.

Jelölje P^*_1, P^*_{2a}, P^*_3, P^*_{2b} a blokkok hosszát a 3 gépes optimumban.

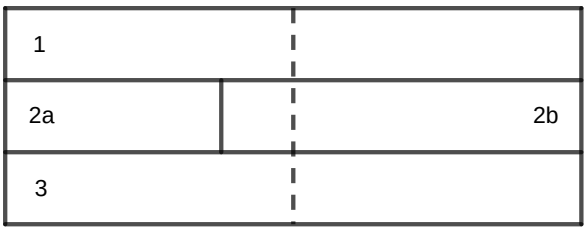
Ekkor a 2 gépes optimumokban a blokkok hosszára a következők teljesülnek:

$$P_1 \leq \max\left\{P^*_1, P^*_{2a} + \frac{2}{3}P_{max}\right\}$$

$$P_{2a} \leq \max\left\{P^*_1 - \frac{2}{3}P_{max}, P^*_{2a}\right\}$$

Algoritmus 3: 5/3-approximáció

$k := argmax\{p(2,j) : j = 1, \ldots, n_2\}$
ha $p(2,k) \geq \frac{1}{3}P_{max}$ *és* $(2,k)$ *nem ütköző vagy* $(2,k) \in U_{1,2}$ **akkor**
  Határozzuk meg az 1. és 2. gép műveleteire a 2 gépes feladat optimumát.
  Az 1. és 2. gépen a $C(2,k)$ időpont után befejeződő műveleteket toljuk el $P_3 - p(2,k)$-val.
  A 3. gépen ütemezzünk szorosan az $[S(2,k), C(2,k) + (P_3 - p(2,k))]$ intervallumban.
ha $p(2,k) \geq \frac{1}{3}P_{max}$ *és* $(2,k) \in U_{2,3}$ **akkor**
  Határozzuk meg a 2. és 3. gép műveleteire a 2 gépes feladat optimumát.
  A 2. és 3. gépen a $C(2,k)$ időpont után befejeződő műveleteket toljuk el $P_1 - p(2,k)$-val.
  Az 1. gépen ütemezzünk szorosan az $[S(2,k), C(2,k) + (P_1 - p(2,k))]$ intervallumban.
ha $p(2,k) \leq \frac{1}{3}P_{max}$ **akkor**
  $C^{alg}_{max} := \infty$
  ciklus *a 2. gép minden 2a-2b kettéosztására*:
    Legyen β egy $\frac{2}{3}P_{max}$ hosszú nem ütköző művelet.
    Határozzuk meg a 2 gépes optimumot az $[1, 2a + \beta]$ esetben.
    Határozzuk meg a 2 gépes optimumot a $[2b \text{ vissza} + \beta, 3 \text{ vissza}]$ esetben.
    A kapott megoldásokat rakjuk össze a 4.2. ábrának megfelelően.
    $C^{alg}_{max} := \min\{C^{alg}_{max}, C_{max}\}$



4.1. ábra. A blokkok meghatározása

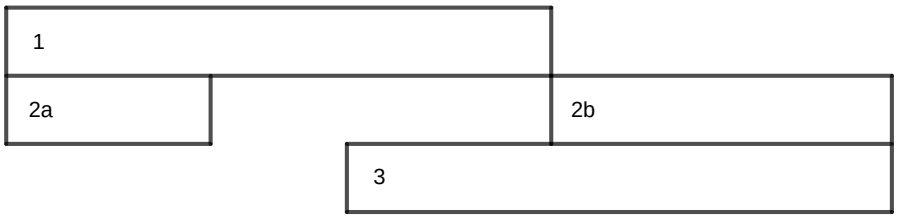
$$P_3 \leq \max\left\{P^*_{2b} + \frac{2}{3}P_{max}, P^*_3\right\}$$

$$P_{2b} \leq \min\left\{P^*_{2b}, P^*_3 - \frac{2}{3}P_{max}\right\}$$

Azt is tudjuk, hogy $C^*_{max} = \max\{P^*_1, P^*_{2a} + P^*_{2b}, P^*_3\}$. Továbbá, ebben az esetben minden művelet feldolgozási idege legfeljebb $\frac{1}{3}P_{max} \leq \frac{1}{3}C^*_{max}$, így $P^*_{2a} \leq (\frac{1}{2} + \frac{1}{6})C^*_{max} = \frac{2}{3}C^*_{max}$ és $P^*_{2a} \leq \frac{2}{3}C^*_{max}$.

A 2 gépre vonatkozó részmegoldásokat úgy rakjuk össze, hogy 1 és 2b, ill. 2a és 3 ne lógjanak egymásba, ld. 4.2. ábra.

A fentiek alapján, és a 4.1.1. lemmát ($P_{max} \leq C^*_{max} \leq 2P_{max}$) felhasználva, a kapott megoldás teljes átfutási idejét a következők szerint tudjuk becsülni.



4.2. ábra. A kapott ütemterv

$$
\begin{aligned}
C_{max} &\leq \max\{P_1 + P_{2b}, P_{2a} + P_3\} \\
&\leq \max\left\{P_1^* + P_{2b}^*, P_1^* + P_3^* - \frac{2}{3}P_{max}, P_{2a}^* + \frac{2}{3}P_{max} + P_{2b}^*, P_{2a}^* + P_3^*\right\} \\
&\leq \max\left\{\left(1 + \frac{5}{3}\right)C_{max}^*, \left(2 - \frac{1}{3}\right)C_{max}^*, \left(1 + \frac{2}{3}\right)C_{max}^*, \left(\frac{2}{3} + 1\right)C_{max}^*\right\} = \frac{5}{3}C_{max}^*
\end{aligned}
$$

Tehát $C_{max}^{alg} \leq C_{max} \leq \frac{5}{3}C_{max}^*$. □

4.3. 11/6-approximáció 4 gépre

Most a $PD4|machine\ chains, \mathcal{U}_{i,i+1}|C_{max}$ feladatra fogunk mutatni egy 11/6-approximációs algoritmust. Ez hasonló lesz, mint az előző részben a 3 gépes feladatra bemutatott 5/3-approximáció, de nem tekinthető annak közvetlen általánosításának.

Ha a 2. vagy 3. gépen van olyan művelet, amely $\frac{1}{6}P_{max}$-nál hosszabb, és nincs benne az $U_{2,3}$ ütköző halmazban, akkor az 5/3-approximációs algoritmus első feléhez hasonlóan meg tudunk adni egy olyan ütemtervet, amelyre $C_{max} \leq \frac{11}{6}C_{max}^*$. Azonban az 5/3-approximációval ellentétben ez a módszer nem csak a „rövid" műveletekből álló példákon nem használható, hanem akkor sem, ha az $\frac{1}{6}P_{max}$-nál hosszabb műveletek $U_{2,3}$-ban vannak. Az ilyen feladatpéldányokat további két csoportba fogjuk osztani. Ezekben a 2 gépes algoritmusból kapott részütemterveket két különböző konfiguráció szerint fogjuk összeállítani. Mindkét konfiguráció esetén mutatunk 1-1 konkrét megoldást, amelyre $C_{max} \leq \frac{11}{6}C_{max}^*$.

Legyenek $(2, i)$ és $(3, j)$ a második, ill. harmadik gép leghosszabb műveletei: $p(2, i) \geq \geq \max\{p(2, k) : k = 1, \ldots, n_2\}$, $p(3, j) \geq \max\{p(3, k) : k = 1, \ldots, n_3\}$.

Ha $p(2, i) \geq \frac{1}{6}P_{max}$ és $(2, i) \notin U_{2,3}$, azaz $(2, i)$ nem ütköző vagy $(2, i) \in U_{1,2}$, akkor határozzuk meg a 2 gépes optimumot külön az 1-2 és a 3-4 gépekre. $C_{max}^{1,2} \leq C_{max}^*$ és $C_{max}^{3,4} \leq C_{max}^*$. Az 1-2 gépeken a $C(2, i)$ után befejeződő műveleteket toljuk el $C_{max}^{3,4} - - p(2, i)$-vel. És a 3-4 gépekre kapott megoldást ütemezzük $S(2, i)$-től kezdve.

A 2. gépen csak a $(2, i) \notin U_{2,3}$ művelet feldolgozása esik az $[S(2, i), C(2, i) + C_{max}^{3,4} - - p(2, i)]$ intervallumba, ezért nem lehet ütközés a 2. és 3. gép között.

A kapott ütemterv teljes átfutási ideje: $C_{max} = C_{max}^{1,2} + C_{max}^{3,4} - p(2, i) \leq (2 - \frac{1}{6})C_{max}^*$.

Ha $p(3, j) \geq \frac{1}{6}P_{max}$ és $(3, j) \notin U_{2,3}$, azaz $(3, j)$ nem ütköző vagy $(3, j) \in U_{3,4}$, akkor az előzőhöz hasonlóan meg tudunk adni egy olyan ütemtervet, amelyre $C_{max} \leq \frac{11}{6}C_{max}^*$.

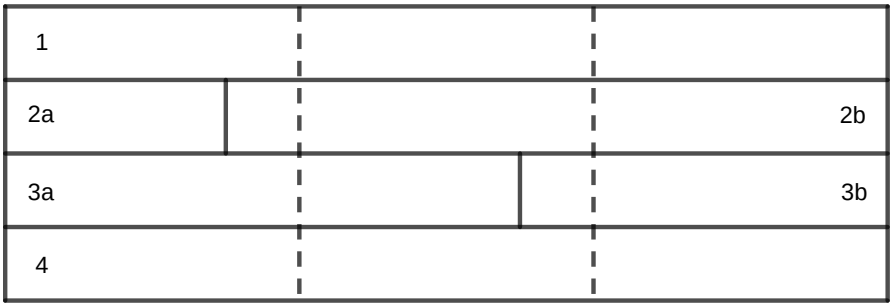
A továbbiakban tehát feltehetjük, hogy a 2. és 3. gépen minden $\frac{1}{6}P_{max}$-nál hosszabb művelete $U_{2,3}$-beli.

A további esetekben a 2. és 3. gép műveleteit kisebb részhalmazokba, blokkokba osztjuk. Az 5/3-approximációhoz hasonlóan különböző módon definiált, új nem ütköző

segédműveleteket is tartalmazó, kisebb 2 gépes feladatokon meghatározzuk az optimális megoldást. Végül ezeket a részütemterveket felhasználva elkészítünk egy megengedett ütemtervet. Két különböző elrendezést is használni fogunk.

Első konfiguráció

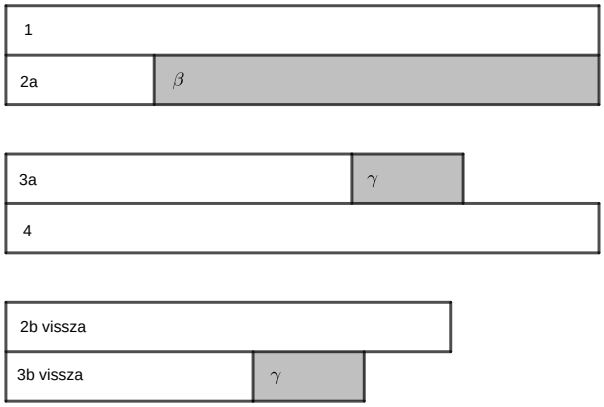
Először az optimális megoldásban a 2. gépet vágjuk el az $\frac{1}{3}C^*_{max}$-hoz legközelebbi művelet-határon, a 3. gépet pedig a $\frac{2}{3}C^*_{max}$-hoz legközelebbi művelet-határon. Így definiáljuk a 2a, 2b, 3a, 3b blokkokat, ld. 4.3. ábra.



4.3. ábra. A blokkok meghatározása az első konfigurációban

Ezt a konfigurációt akkor használjuk, ha a meghatározott vágás helyének a megfelelő harmadolóponttól vett távolsága mindkét gép esetén legfeljebb $\frac{1}{12}C^*_{max}$. Tehát az optimumban a blokkok hosszára teljesül, hogy $P^*_{2a} \leq \frac{5}{12}, P^*_{2b} \leq \frac{3}{4}, P^*_{3a} \leq \frac{3}{4}, P^*_{3b} \leq \frac{5}{12}$.

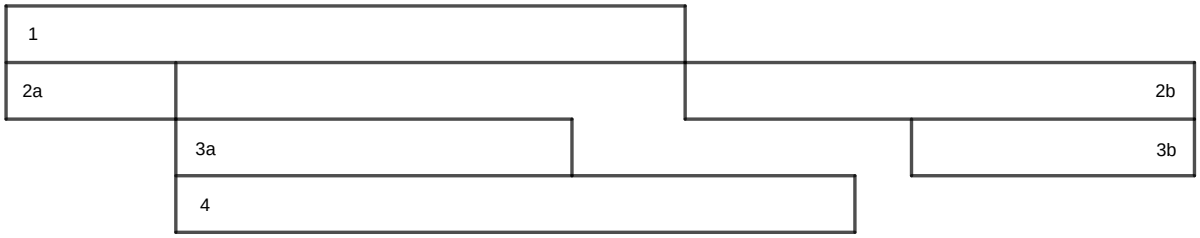
Határozzuk meg a 2 gépes optimumot az [1, 2a + β], [3a + γ, 4] és [2b vissza, 3b vissza + γ] esetekben, ahol β, ill. γ nem ütköző műveleteket jelölnek, ld. 4.4. ábra. Az így kapott megoldásokban a blokkok hossza legyen $P_1, P_{2a}, P_{2b}, P_{3a}, P_{3b}, P_4$.



4.4. ábra. A 2 gépes feladat inputjai az első konfigurációban

A kapott részmegoldásokat rakjuk össze. 1 és 2a kezdődjön a 0 időpontban, 3a és 4 pedig a 2a befejezésével egyidőben. Végül 2b-t és 3b-t toljuk el együtt a végére úgy, hogy 2b az 1 és 3a után kezdődjön, 3b pedig a 4 után, ld. 4.5. ábra. Az így kapott ütemtervben nem lehet ütközés, mert különböző 2 gépes feladatokból jövő műveleteket nem ütemezünk egy időben.

Ekkor $C_{max} = \max\{P_1 + P_{2b}, P_{2a} + P_{3a} + P_{2b}, P_{2a} + P_4 + P_{3b}\}$. Ez legfeljebb annyi, mint az alábbi sorok maximuma. A célunk a $\frac{11}{6}$-approximáció, ezért minden sor végére odaírtam, hogy mit kell teljesítenie β-nak és γ-nak ahhoz, hogy $\leq \frac{11}{6}C^*_{max}$-ot kapjunk.



4.5. ábra. Az első konfigurációból kapott ütemterv

$P_1^* + P_{2b}^*$	$\leq (1+\frac{3}{4})C_{max}^*$	
$P_1^* + P_{3b}^* + \gamma$	$\leq (1+\frac{5}{12})C_{max}^* + \gamma$	$\gamma \leq \frac{5}{12}C_{max}^*$
$P_{2a}^* + \beta + P_{2b}^*$	$= C_{max}^* + \beta$	$\beta \leq \frac{5}{6}C_{max}^*$
$P_{2a}^* + \beta + P_{3b}^* + \gamma$	$\leq 2 \cdot \frac{5}{12}C_{max}^* + \beta + \gamma$	$\beta + \gamma \leq 1$
$P_{2a}^* + P_{3a}^* + P_{2b}^*$	$\leq (1+\frac{3}{4})C_{max}^*$	
$P_{2a}^* + P_{3a}^* + P_{3b}^* + \gamma$	$\leq (\frac{5}{12}+1)C_{max}^* + \gamma$	$\gamma \leq \frac{5}{12}C_{max}^*$
$P_{2a}^* + P_4^* - \gamma + P_{2b}^*$	$= 2C_{max}^* - \gamma$	$\gamma \geq \frac{1}{6}C_{max}^*$
$P_{2a}^* + P_4^* + P_{3b}^*$	$\leq (\frac{5}{12}+1+\frac{5}{12})C_{max}^*$	
$P_1^* - \beta + P_{3a}^* + P_{2b}^*$	$\leq (1+2\cdot\frac{3}{4})C_{max}^* - \beta$	$\beta \geq \frac{2}{3}C_{max}^*$
$P_1^* - \beta + P_{3a}^* + P_{3b}^* + \gamma$	$= 2C_{max}^* - \beta + \gamma$	$\beta - \gamma \geq \frac{1}{6}C_{max}^*$
$P_1^* - \beta + P_4^* - \gamma + P_{2b}^*$	$\leq (2+\frac{3}{4})C_{max}^* - \beta - \gamma$	$\beta + \gamma \geq \frac{11}{12}C_{max}^*$
$P_1^* - \beta + P_4^* + P_{3b}^*$	$\leq (2+\frac{5}{12})C_{max}^* - \beta$	$\beta \geq \frac{7}{12}C_{max}^*$

Ezek alapján a β-ra és γ-ra vonatkozó feltételek:

$$\frac{2}{3}C_{max}^* \leq \beta \leq \frac{5}{6}C_{max}^*$$

$$\frac{1}{6}C_{max}^* \leq \gamma \leq \frac{5}{12}C_{max}^*$$

$$\frac{11}{12}C_{max}^* \leq \beta + \gamma \leq C_{max}^*$$

Az egyszerűség kedvéért tegyük fel, hogy $\beta = 4\gamma$. Így az egyetlen feltétel az marad, hogy $\frac{11}{15}C_{max}^* \leq \beta \leq \frac{4}{5}C_{max}^*$.

A 4.1.1. lemmából tudjuk, hogy $P_{max} \leq C_{max}^* \leq 2P_{max}$, azaz $P_{max} \in \left[\frac{1}{2}C_{max}^*, C_{max}^*\right]$.

Bontsuk a feladatod 8 esetre, a k-adik esetben $P_{max} \in \left((\frac{11}{12})^k C_{max}^*, (\frac{11}{12})^{k-1} C_{max}^*\right]$ ($k = 1, \ldots, 8$). Mivel $(\frac{11}{12})^0 = 1$ és $(\frac{11}{12})^8 < \frac{1}{2}$, így ezekkel lefedjük a fenti intervallumot.

A k-adik esetben $\frac{11}{15}C_{max}^* \leq \frac{11}{15}(\frac{12}{11})^k P_{max} \leq \frac{4}{5}C_{max}^*$.

Legyen $\beta_k = \frac{11}{15}(\frac{12}{11})^k P_{max}$, $\gamma_k = \frac{1}{4}\beta_k$ ($k = 1, \ldots, 8$).

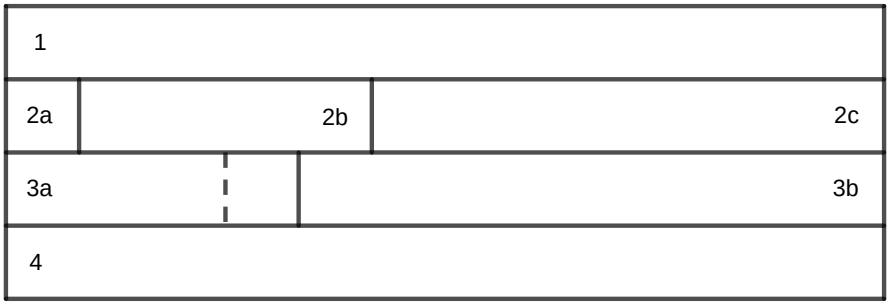
Az algoritmusban határozzuk meg a megoldást minden 2a-2b, 3a-3b szétosztásra, és mind a 8 lehetséges β_k és γ_k értékre. Ezek között biztosan megvizsgáljuk azt a szétbontást, amit az optimumból kaphatnánk, és valamelyik β_k, γ_k pár teljesíteni fogja az összes fenti feltételt, így a meghatározott megoldások közül a legkisebbre biztosan igaz, hogy $C_{max} \leq \frac{11}{6}C_{max}^*$.

Második konfiguráció

A fenti elrendezést pontosan akkor nem tudjuk használni, ha a 2. gépen van egy $(2, i) \in U_{2,3}$ művelet, amelyre $S^*(2, i) < \frac{1}{4}C_{max}^*$ és $C^*(2, i) > \frac{5}{12}C_{max}^*$, vagy a 3. gépen van egy

$(3, j) \in U_{2,3}$ művelet, amelyre $S^*(3, j) < \frac{7}{12}C^*_{max}$ és $C^*(3, j) > \frac{3}{4}C^*_{max}$, mert ekkor a 4.3. ábrán a harmadolópontoktól túl távol kerülnek a vágások. Szimmetria miatt ezek közül elég az elsőt megvizsgálni.

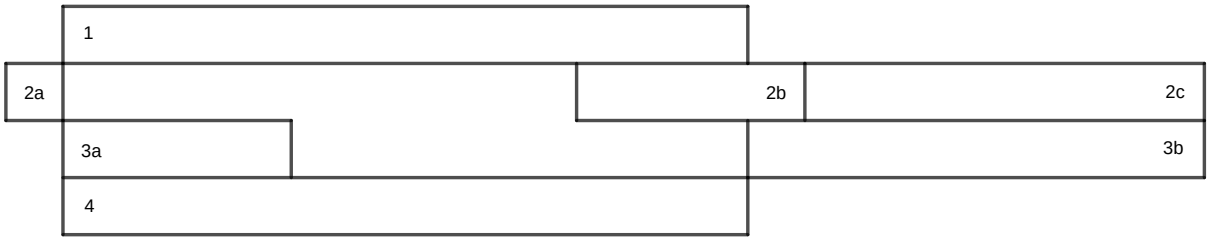
A 2. gépet három részre bontjuk, a 3. gépet pedig két részre. Az optimális megoldásban a B blokk hosszát a korábbiakhoz hasonlóan P^*_B-gal fogjuk jelölni. A 2b blokk álljon önmagában a $(2, i)$ műveletből. 2a, ill. 2c pedig legyenek az előtte, ill. utána lévő műveletekből álló blokkok. 3a-t úgy választjuk, hogy 2a befejezése után legalább $\frac{1}{6}C^*_{max}$-gal érjen véget, azaz $P^*_{3a} \geq P^*_{2a} + \frac{1}{6}C^*_{max}$. Mivel $P^*_{2b} \geq \frac{1}{6}$, ezért ebben az időpontban a 2. gépen a 2a művelet van feldolgozás alatt. Így a 3. gépen ebben az időpontban nem lehet $U_{2,3}$-beli művelet, tehát az épp feldolgozott művelet legfeljebb $\frac{1}{6}C^*_{max}$ hosszú, és így $P^*_{3a} \leq P^*_{2a} + \frac{1}{3}$.



4.6. ábra. A blokkok meghatározása a második konfigurációban

Jelölje P_B a B blokk hosszát az alábbiakban leírt szabály szerint „több részből összerakott” megoldásban, és legyen $\hat{P}_B$ a B blokk hossza, ha műveleteket szorosan ütemezzük.

2a és 1 műveleteit rakjuk szorosan egymás után, azaz $P_{2a} = \hat{P}_{2a}$, $P_1 = \hat{P}_1$. Ezen kívül határozzuk meg a 2 gépes optimumot a [3a + γ, 4] és [2c vissza + 2b, 3b vissza + γ] esetekben, ahol γ egy nem ütköző művelet. 1, 3a és 4 kezdődjön a 2a befejezésével egyidőben. 2b-t, 2c-t és 3b-t együtt toljuk el úgy, hogy 2b a 3a után kezdődjön, 2c az 1 után, és 3b a 4 után, ld. 4.7. ábra.



4.7. ábra. A második konfigurációból kapott ütemterv

A 2. gép műveletei közül csak a $2b \notin U_{1,2}$ lehet olyan, hogy az 1. gép műveletivel egy időben van ütemezve, tehát az 1-2. gépek között nem lehet ütközés. A 2. és 3. gépen csak a 2b, 2c, 3b műveletei lehetnek egy időben, de ezeket a 2 gépes feladat szerint úgy ütemezzük, hogy ne legyen ütközés. És hasonlóan a 3. és 4. gépek esetében a két gépes feladat megoldása miatt a 3a és 4 blokkok között sem lehet ütközés.

$$
\begin{aligned}
C_{max} &= \max\{P_{2a}+P_4+P_{3b}, P_{2a}+P_{3a}+P_{2b}+P_{2c}, P_{2a}+P_1+P_{2c}\} \\
&\le \max\Big\{P^*_{2a}+P^*_4+P^*_{3b}, P^*_{2a}+P^*_4+P_{2b}+P^*_{2c}-\gamma, \hat{P}_{2a}+P^*_{3a}+\gamma+P^*_{3b}, \\
&\qquad P^*_{2a}+P^*_{3a}+P_{2b}+P^*_{2c}, P^*_{2a}+P^*_1+P^*_{2c}, P^*_{2a}+P^*_1+P^*_{3b}+\gamma-P_{2b}\Big\} \\
&\le \max\left\{\frac{11}{6}C^*_{max}, 2C^*_{max}-\gamma, C^*_{max}+\hat{P}_{2a}+\gamma, \frac{19}{12}C^*_{max}, \frac{11}{6}C^*_{max}, \frac{11}{6}C^*_{max}+\gamma-P_{2b}\right\}
\end{aligned}
$$

Ez $\le \frac{11}{6}C^*_{max}$, ha $\frac{1}{6}C^*_{max} \le \gamma \le \min\left\{\frac{5}{6}C^*_{max}-\hat{P}_{2a}, P_{2b}\right\}$.

1. eset: Ha $\hat{P}_{2a}+P_{2b} \le \frac{5}{6}C^*_{max}$, akkor $\gamma = P_{2b}$ jó.

2. eset: Ha $\hat{P}_{2c}+P_{2b} \le \frac{5}{6}C^*_{max}$, akkor nézhetjük a feladat „függőlegesen tükrözött" változatát, amiben a 2a helyett a 2c-t ütemezzük szorosan, a 3a, 3b választását is megfelelően módosítjuk, stb. Ebben az esetben a γ-ra vonatkozó feltétel az lesz, hogy $\frac{1}{6}C^*_{max} \le \gamma \le$ $\le \min\left\{\frac{5}{6}C^*_{max}-\hat{P}_{2c}, P_{2b}\right\}$. És így a $\gamma = P_{2b}$ választás jó lesz.

3. eset: Különben $\hat{P}_{2c} \le \frac{1}{6}C^*_{max}$ és $\hat{P}_{2a} \le \frac{1}{6}C^*_{max}$, tehát $P_{2b} \ge \frac{2}{3}C^*_{max}$. Ekkor a γ-ra vonatkozó feltétel: $\frac{1}{6}C^*_{max} \le \gamma \le \frac{2}{3}C^*_{max}$. A 4.1.1. lemma alapján $\frac{1}{3}C^*_{max} \le \frac{2}{3}P_{max} \le$ $\le \frac{2}{3}C^*_{max}$, tehát $\gamma = \frac{2}{3}P_{max}$ jó lesz.

A 3. gép minden 3a-3b kettéosztását vizsgáljuk meg. A 2b választásánál viszont figyelembe vehetjük az alábbiakat. Az 1. és 2. esetben elég az olyan $U_{2,3}$-beli műveletekre mint 2b-re kiszámolni a fentieket, amire $P_{2b} \ge \frac{1}{6}P_{max}$. A 3. esetben pedig elég azt nézni, ahol $P_{2b} \ge \frac{2}{3}P_{max}$. Továbbá az 1. és 2. eset közül elég csak az egyiket megvizsgálni aszerint, hogy $\hat{P}_{2a}$ vagy $\hat{P}_{2c}$ a kisebb.

Ha az ezeknek megfelelő összes 2a-2b-2c, 3a-3b blokkfelbontásra, és $\gamma = P_{2b}$-re és $\gamma = \frac{2}{3}P_{max}$-re is meghatározzuk a 4.7. ábrának megfelelő ütemtervet, akkor ezek között megvizsgáljuk azt az esetet is, amelyet az optimális ütemtervből kaphatnánk. Ezért a legjobb ütemterv legfeljebb olyan hosszú, mint amelyikről beláttuk, hogy legfeljebb $\frac{11}{6}C^*_{max}$ hosszú.

A fentiek alapján felírhatjuk a 4. algoritmust, és kimondhatjuk a következő tételt:

4.3.1. Tétel. *A 4. algoritmus 11/6-approximáció a* $PD4|machine\ chains,\mathcal{U}_{i,i+1}|C_{max}$ *feladatra.*

Algoritmus 4: 11/6-approximáció

ha $\exists i : p(2,i) \geq \frac{1}{6}P_{max}$ *és* $(2,i)$ *nem ütköző vagy* $(2,i) \in U_{1,2}$ **akkor**
  Határozzuk meg a 2 gépes optimumot az 1-2, ill. a 3-4 gépre.
  Az 1. és 2. gépen a $C(2,i)$ időpont után befejeződő műveleteket toljuk el $C^{3,4}_{max} - p(2,i)$-vel.
  A 3-4 gépekre kapott megoldást ütemezzük az $[S(2,i), C(2,i) + (C^{3,4}_{max} - p(2,i))]$ intervallumba.
különben ha $\exists i : p(3,i) \geq \frac{1}{6}P_{max}$ *és* $(3,i)$ *nem ütköző vagy* $(3,i) \in U_{3,4}$ **akkor**
  Határozzuk meg a 2 gépes optimumot az 1-2, ill. a 3-4 gépre.
  A 3. és 4. gépen a $C(3,i)$ időpont után befejeződő műveleteket toljuk el $C^{1,2}_{max} - p(3,i)$-vel.
  Az 1-2 gépekre kapott megoldást ütemezzük az $[S(3,i), C(3,i) + (C^{1,2}_{max} - p(3,i))]$ intervallumba.
különben
  $C^{alg}_{max} := \min\{$ `ElsoKonfig(`*1, 2, 3, 4*`)`, `MasodikKonfig(`*1, 2, 3, 4*`)`, `MasodikKonfig(`*4, 3, 2, 1*`)` $\}$

Függvény ElsoKonfig(1, 2, 3, 4)

$C^{alg}_{max} := \infty$
ciklus *a 2. gép minden 2a-2b kettéosztására*:
  ciklus *a 3. gép minden 3a-3b kettéosztására*:
    for $k = 1, \ldots, 8$:
      Legyen β egy $\frac{11}{15}(\frac{12}{11})^k P_{max}$ hosszú nem ütköző művelet.
      Legyen γ egy $\frac{1}{4}\beta$ hosszú nem ütköző művelet.
      Határozzuk meg a 2 gépes optimumot az $[1, 2a + \beta]$ esetben.
      Határozzuk meg a 2 gépes optimumot a $[3a + \gamma, 4]$ esetben.
      Határozzuk meg a 2 gépes optimumot az $[2b \text{ vissza}, 3b \text{ vissza} + \gamma]$ esetben.
      A kapott megoldásokat rakjuk össze a 4.5. ábrának megfelelően.
      $C^{alg}_{max} := \min\{C^{alg}_{max}, C_{max}\}$
return C^{alg}_{max}

Függvény MasodikKonfig(1, 2, 3, 4)

```
1  C^alg_max := ∞
2  ciklus 2b ∈ {(2,i) ∈ U_{2,3} : p(2,i) ≥ 1/6 P_max}:
3      2a és 2c legyenek a 2b előtti, ill. utáni műveletekből álló blokkok.
4      ciklus a 3. gép minden 3a-3b kettéosztására:
5          Legyen γ egy P_2b hosszú nem ütköző művelet.
6          ha P̂_2a ≤ P̂_2c akkor
7              C_max := MasodikKonfigUtemterv(1, 2a, 2b, 2c, 3a, 3b, 4, γ)
8              C^alg_max := min{C^alg_max, C_max}
9          különben
10             C_max := MasodikKonfigUtemterv(1, 2c, 2b, 2a, 3b, 3a, 4, γ)
11             C^alg_max := min{C^alg_max, C_max}
12         ha P_2b ≥ 2/3 P_max akkor
13             Legyen γ egy 2/3 P_max hosszú nem ütköző művelet.
14             C_max := MasodikKonfigUtemterv(1, 2a, 2b, 2c, 3a, 3b, 4, γ)
15             C^alg_max := min{C^alg_max, C_max}
16 return C^alg_max
```

Függvény MasodikKonfigUtemterv(1, 2a, 2b, 2c, 3a, 3b, 4, γ)

```
1  A 2a és 1 műveleteit ütemezzük szorosan.
2  Határozzuk meg a 2 gépes optimumot a [3a + γ, 4] esetben.
3  Határozzuk meg a 2 gépes optimumot a [2c vissza + 2b, 3b vissza + γ] esetben.
4  A kapott megoldásokat rakjuk össze a 4.7. ábrának megfelelően.
5  return C_max
```

5. fejezet

Összegzés

Dolgozatomban a speciális ütközési feltételeket és megelőzési korlátokat tartalmazó $PD|machine\ chains,\mathcal{U}_{i,i+1}|C_{max}$ ütemezési feladattal foglalkoztam. Ebben a műveletek már előre hozzá vannak rendelve a gépekhez, és ezek sorrendje is adott. Továbbá meg vannak határozva olyan ütköző művelethalmazok, melyek két szomszédos gép műveleteit tartalmazzák. A feladat azon változatára, amelyben a gépek száma 2, két különböző algoritmust is bemutattam, amelyek segítségével meghatározható az optimális ütemterv. Az első algoritmus dinamikus programozást használt, a másik pedig egy geometriai ötletre épült, és irányított gráfban történő legrövidebb útkeresés segítségével oldotta meg a problémát.

Ezután bebizonyítottam, hogy ha három gép van, akkor a feladat NP-nehéz. A bizonyításban a páros-páratlan partíció feladatot vezettük vissza az ütemezési problémára. Három vagy több gép esetén tehát nem várható, hogy polinom időben meghatározható az optimális megoldás. Emiatt közelítő algoritmusokat adtam a feladatra. A bemutatott 2-közelítő algoritmus tetszőleges számú gép esetén használható. Ezen kívül adtam egy 5/3-approximációt 3 gépre és egy 11/6-approximációt 4 gépre.

A vizsgált ütemezési feladatok nagyon hasonlítanak a job shop problémára. Ezt kihasználva új bizonyítást is adtam az ismert tételre, miszerint a $J3|n = 3|C_{max}$ job shop feladat NP-nehéz.

A többfejes épületnyomtatásnál megköveteltük, hogy az ütközőzónában egyszerre csak egy nyomtatófej lehessen. Az ütemezési feladatban ez annak felelt meg, hogy az azonos halmazba eső ütköző műveletekről megköveteltük, hogy diszjunktak legyenek. Azonban két nyomtatófej akkor sem ütközik össze, ha a távolságuk elég nagy, de mindketten az ütköző zónában vannak. Ha az egyik fej már kifelé halad az ütköző zónából, akkor a másiknak nem kell megvárnia, hogy teljesen kiérjen belőle. Ha betartjuk a megfelelő távolságot a két fej között, akkor a második fej már korábban is folytathatja az útját az ütköző zónába. Ezzel a várakozási időket lecsökkenthetjük, és így az egész nyomtatási folyamat ideje is lerövidülhet. Ezt felhasználva érdemes lehet megvizsgálni, hogy ki lehet-e dolgozni még hatékonyabb módszereket az ütközés elkerülésére.

A feladatok megoldásánál feltettük, hogy tetszőleges helyre beszúrhatunk várakozást, és ennek időtartamára sem volt semmilyen korlátunk. A gyakorlatban azonban az anyag áramlásának sorozatos leállítása és újraindítása, és a nyomtatófej gyakori sebességváltozása a felületek egyenlőtlenségéhez vezethet [2]. A nyomtatás minőségének javítása érdekében ezeket a szempontokat is be kell építeni a megoldási modellekbe.

Elméleti szempontból érdekes lehetne a vizsgált ütemezési feladatokat úgy is tanulmányozni, hogy a C_{max} teljes átfutási idő helyett más célfüggvényeket használunk.

Az is egy érdekes feladat lehet, hogy az ütközéses ütemezési feladatra olyan 2-nél jobb közelítést adó, esetleg PTAS algoritmust adjunk, amely tetszőleges számú gép esetén használható. Vagy negatív eredményként bebizonyítsuk, hogy egy bizonyos konstansnál jobban nem közelíthető a feladat.

Irodalomjegyzék

[1] Jing Zhang, Behrokh Khoshnevis, *Optimal machine operation planning for construction by Contour Crafting*, Automation in Construction 29 (2013) 50-67

[2] Yuan Jin, Jianke Du, Zhiyong Ma, Anbang Liu, Yong He, *An optimization approach for path planning of high-quality and uniform additive manufacturing*, The International Journal of Advanced Manufacturing Technology (2017) 92:651-662

[3] Contour Crafting Corporation honlapja: http://contourcrafting.com/

[4] S. Lin, B. Kernighan, *An effective heuristic algorithm for the traveling salesman problem*, Operations Research 21 (1973) 498–516.

[5] Jordán Tibor: *Ütemezés* (2007)

[6] H-C. Hong, B.M.T. Lin, *Parallel Dedicated Machine Scheduling with Conflict Graphs*, Computers & Industrial Engineering (2018)

[7] A. Agnetis, H. Kellerer, G. Nicosia, A. Pacifici, (2012) *Parallel dedicated machines scheduling with chain precedence constraints.* European Journal of Operational Research, 221(2), 296-305.

[8] P. Brucker, Y. N. Sotskov, F. Werner, (2007). *Complexity of shop-scheduling problems with fixed number of jobs: a survey.* Mathematical Methods of Operations Research, 65(3), 461-481.

[9] Yu.N. Sotskov, N.V. Shakhlevich, *NP-hardness of shop-scheduling problems with three jobs*, Discrete Applied Mathematics 59 (1995) 237-266