Miskolci Egyetem
Gépészmérnöki és Informatikai Kar
Informatikai Intézet
Alkalmazott Informatikai Intézeti Tanszék

ET Erőforrás tervezés
Resource Planning

2017/18 2. félév

3. rész

Dr. Kulcsárné Dr. Forrai Mónika egyetemi adjunktus
Dr. Kulcsár Gyula egyetemi docens

Erőforrás tervezés

Egygépes ütemezési feladatok

Egygépes ütemezési feladatok

Az egy erőforrást (gépet) tartalmazó ütemezési feladatok megoldásával foglalkozunk.

A nagyszámú feladat közül *olyan példák*at válogattunk össze, *amelyek megoldásához szükséges algoritmusok legfeljebb polinomiális futási idejűek*.

Egygépes ütemezési feladatok

Feltételezzük, hogy:

- az erőforrás az ütemezési időszakban folyamatosan rendelkezésre áll,
- az erőforrás egyszerre csak egy munkán dolgozhat,
- a munkák legkorábbi indítási időpontja nulla: $r_i = 0$ (i=1, 2, …, n),
- minden egyes munkához egyetlen operáció tartozik, melyeknek pontosan ismert a végrehajtási ideje: p_i (i=1, 2, …, n),
- az operációk végrehajtása nem szakítható meg.

Egygépes ütemezési feladatok

Az egygépes esetben a lehetséges ütemtervek száma: n! (n faktoriális).

Egy lehetséges megoldás (permutáció) egy n elemű vektorral reprezentálható. A vektor elemeinek sorszáma mutatja a munka végrehajtási sorban elfoglalt helyét.

Példa:

Sorszám	1	2	3	4
Munka indexe	1	3	4	2

Végrehajtási sorrend: $J_1 \rightarrow J_3 \rightarrow J_4 \rightarrow J_2$

Az SPT szabály alkalmazása

A feladat formális leírása: $1||\sum C_i$

Használható az $1||C_{\text{sum}}$ jelölés is.

A feladat értelmezése: Egyetlen erőforráson adott számú munkát kell elvégezni.

A tanult alapértelmezett végrehajtási jellemzők vannak érvényben.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely mellett a munkák befejezési időpontjainak összege minimális lesz.

Az SPT szabály alkalmazása

A feladat megoldása:

A "legrövidebb műveleti idejű munka előre" (Shortest Processing Time, SPT) szabály az $1||\sum C_i$ feladat optimális megoldását adja.

Az SPT szabály alkalmazása:

Rendezzük a munkákat a műveleti idők alapján nemcsökkenő sorrendbe és ennek megfelelően indítsuk el azokat.

Az SPT szabály alkalmazása

Az átlagos készletszint

N_q a folyamatban lévő munkák számának időegységre vonatkoztatott átlagértéke:

$$N_a = \frac{\sum_{i=1}^{n} C_i}{C_{max}}$$

Emlékeztető: n jelöli a munkák számát, C_i jelöli a J_i munka befejezési időpontját, C_{max} pedig az utolsó munka befejezési időpontjának szimbóluma.

A szakirodalomban elterjedten használják a WIP rövidítést is, ami az adott időpillanatban folyamatban lévő (még be nem fejezett) munkák számát jelenti (Work In Process).

Az SPT szabály alkalmazása

Feladat: Oldjuk meg az $1||\sum C_i$ ütemezési feladatot!

Alapadatok:

i	p_i
1	5
2	3
3	8
4	2

Megoldás: Rendezzük a munkákat SPT sorrendbe!

Pozíció	1	2	3	4
i	4	2	1	3

Az SPT szabály alkalmazása

Az ütemterv Gantt diagramja:

J_4	J_2	J_1	J_3
$C_4 = 2$	$C_2 = 5$	$C_1 = 10$	$C_3 = 18$

célfüggvény értéke:

$$\gamma = C_{sum} = \sum_{i=1}^{n} C_i$$

$$= C_1 + C_2 + C_3 + C_4 = 2 + 5 + 10 + 18 = 35$$

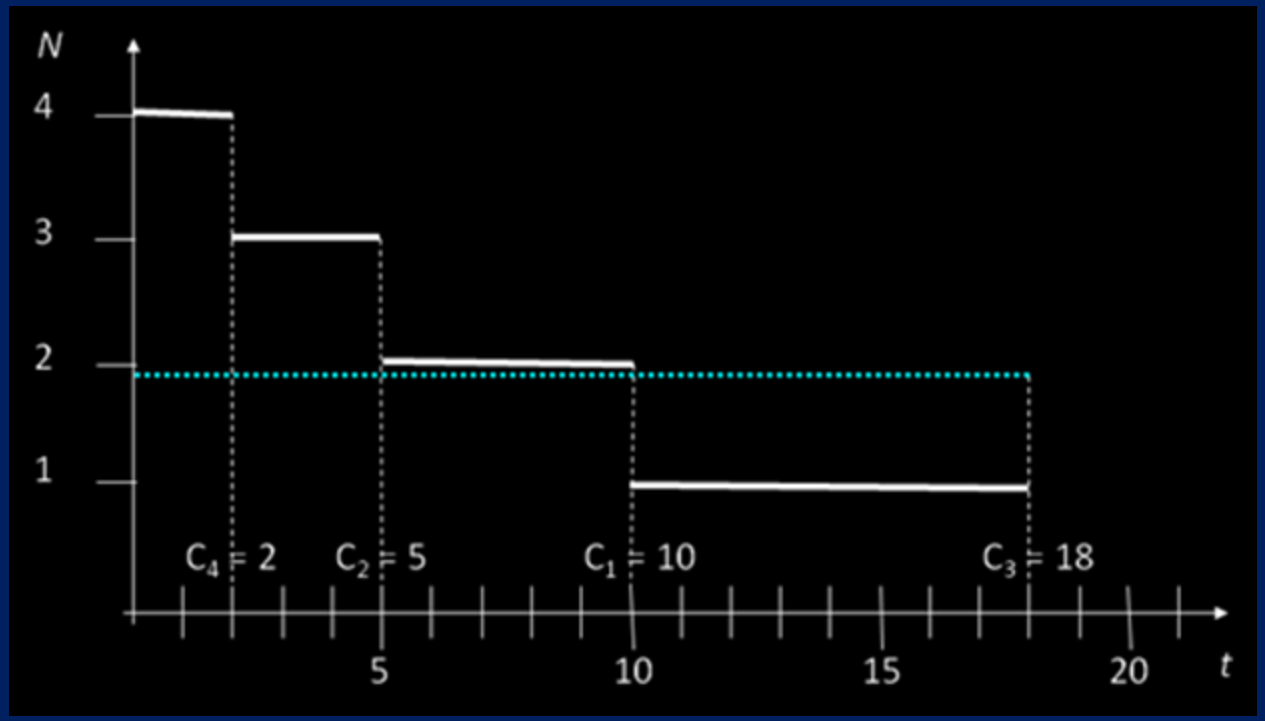
Az SPT szabály alkalmazása

Ábrázoljuk a megoldott feladat készlet-diagramját és számítsuk ki az átlagos készletszint értékét!

Megoldás:

Kezdetben minden munka a várakozó sorban helyezkedik el. Az első munka indítását követően három munka marad a tárolóban és egy a gépre kerül. Amint a munka befejeződik kikerül a rendszerből, vagyis a készletszint diagram értéke eggyel csökken. A következő munka a tárolóból a gépre kerül és ez addig folytatódik, míg a munkák el nem fogynak.

Az SPT szabály alkalmazása



A diagramról leolvashatóak a munkák befejezési időpontjai.

Az SPT szabály alkalmazása

A munkák (diagramról leolvasott) befejezési időpontjait behelyettesítjük a tanult képletbe:

$$N_a = \frac{\sum_{i=1}^{n} C_i}{C_{max}}$$

$$= \frac{C_1 + C_2 + C_3 + C_4}{C_3}$$

$$= \frac{10+5+18+2}{18} = \frac{35}{18}$$

$$Na \approx 1{,}94$$

A WSPT szabály alkalmazása

Oldjuk meg következő ütemezési feladatot: $\mathbf{1}||\sum(\boldsymbol{w_i C_i})$

A feladat értelmezése:

Egyetlen erőforráson adott számú munkát kell elvégezni.

A tanult alapértelmezett végrehajtási jellemzők vannak érvényben.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely mellett a munkák befejezési időpontjainak súlyozott összege minimális lesz.

A WSPT szabály alkalmazása

A feladat megoldása:

A "legkisebb súlyozott műveleti idejű munka előre" (Weighted Shortest Processing Time, WSPT) szabály az $1||\sum(w_i C_i)$ feladat optimális megoldását adja.

A WSPT szabály alkalmazása:

Képezzük minden egyes munka esetében a w_i/p_i hányadosokat, ahol w_i a J_i munka súlyát, a p_i a J_i munka műveleti idejét jelenti, Majd rendezzük a munkákat a kapott w_i/p_i hányadosok alapján nemnövekvő sorrendbe és ennek megfelelően indítsuk el azokat.

A WSPT szabály alkalmazása

Feladat: Oldjuk meg az $1||\sum(w_i C_i)$ ütemezési feladatot!

Alapadatok:

i	p_i	w_i
1	5	2
2	3	6
3	8	5
4	2	3

Megoldás: Számítsuk ki minden egyes munka fontossági mutatóját w_i/p_i!

i	w_i/p_i
1	0,4
2	2
3	0,625
4	1,5

Rendezzük a munkákat WSPT sorrendbe, vagyis a w_i/p_i értékek alapján nemnövekvő sorrendbe:

Pozíció	1	2	3	4
i	2	4	3	1

A WSPT szabály alkalmazása

Az ütemterv Gantt-diagramja:

A célfüggvény értéke:

$$\gamma = \sum(w_i C_i) = \sum_{i=1}^{n} (w_i C_i)$$

$$= w_1 C_1 + w_2 C_2 + w_3 C_3 + w_4 C_4 = 2 \cdot 18 + 6 \cdot 3 + 5 \cdot 13 + 3 \cdot 5 = 134$$

Az EDD szabály alkalmazása

A feladat formális leírása: $1|d_i|T_{max}$

A feladat értelmezése: Egyetlen erőforráson adott számú munkát kell elvégezni.

A tanult alapértelmezett végrehajtási jellemzőkön túl azt is figyelembe kell venni, hogy minden egyes munkának saját dedikált határideje van.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely mellett a munkák legnagyobb csúszása (határidő túllépése) minimális lesz.

Az EDD szabály alkalmazása

$\gamma = max_{i=1}^{n} \{T_i\} \rightarrow \min.$

Megjegyzés: A feladat a maximális késés minimalizálásaként is megfogalmazható: $1|d_i|L_{max}$

A feladat megoldása:

A "legkorábbi határidejű munka előre" (Earliest Due Date, EDD) szabály az $1|d_i|T_{max}$ feladat optimális megoldását adja.

Az EDD szabály alkalmazása:

Rendezzük a munkákat a határidő alapján nemcsökkenő sorrendbe és ennek megfelelően indítsuk el azokat.

Az EDD szabály alkalmazása

Oldjuk meg az $1|d_i|T_{max}$ ütemezési feladatot!

Alapadatok:

i	p_i	d_i
1	5	6
2	3	8
3	8	20
4	2	4

Megoldás: Rendezzük a munkákat EDD sorrendbe, vagyis a határidők alapján nemcsökkenő sorrendbe:

Pozíció	1	2	3	4
i	4	1	2	3

Az ütemterv Gantt-diagramja:

J_4	J_1	J_2	J_3
$C_4 = 2$	$C_1 = 7$	$C_2 = 10$	$C_3 = 18$

Az EDD szabály alkalmazása

Foglaljuk táblázatba az eredményeket az ütemezési sorrend szerint:

i	p_i	d_i	C_i	L_i	T_i
4	2	4	2	-2	0
1	5	6	7	1	1
2	3	8	10	2	2
3	8	20	18	-2	0

A célfüggvény értéke (a maximális csúszás): $\gamma = T_{max} = max_{i=1}^{n}\{T_i\}$

$= \max\{T_1, T_2, T_3, T_4\} = \max\{1,2,0,0\} = 2$

A maximális késés: $L_{max} = max_{i=1}^{n}\{L_i\}$

$= \max\{L_1, L_2, L_3, L_4\} = \max\{1,2,-2,-2\} = 2$

Megjegyzés: Ha van csúszás ($T_{max} > 0$), akkor $T_{max} = L_{max}$, egyébként $T_{max} = 0$, de L_{max} lehet negatív értékű is (minden munka határidő előtt fejeződik be)!

A Moor-algoritmus alkalmazása

A feladat formális leírása: $1|d_i|\sum U_i$

A feladat értelmezése: Egyetlen erőforráson adott számú munkát kell elvégezni. A tanult alapértelmezett végrehajtási jellemzőkön túl azt is figyelembe kell venni, hogy minden egyes munkának saját dedikált határideje van.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely mellett a határidőt túllépő munkák száma minimális lesz.

$$\gamma = max_{i=1}^{n}\{U_i\} \rightarrow \min.$$

A Moor-algoritmus alkalmazása

Jelölje S a munkák ütemezett halmazát, és t az aktuális időpontot.

A feladat megoldását végző Moor-algoritmus formális leírása:

Rendezzük a munkákat EDD szerint: $d_1 \leq d_2 \leq \ldots \leq d_n$;

```
S := φ;   //ures halmaz
t := 0;
FOR i := 1 to n DO
BEGIN
S := S ∪ {i};
t := t + p_i;
IF t > d_i THEN
   BEGIN
    Válasszuk ki azt a j munkát az S halmazból, amelyiknek legnagyobb a p_j értéke;
    S := S\{j};
    t := t − p_j;
   END
END
```

A Moor-algoritmus alkalmazása

A bemutatott algoritmus az $1|d_i|\sum U_i$ feladat optimális megoldását adja.

Az algoritmus értelmezése: Az ütemterv építését elölről kezdjük a munkák EDD szerinti sorrendje alapján.

A munkákat egyesével rakjuk be az ütemezési vektorba balról jobbra haladva.

Ha a beillesztett munka befejezési időpontja meghaladja a munka határidejét, akkor válasszuk ki a már beütemezett munkák közül a legnagyobb műveleti idejűt és vegyük ki az ütemezési vektorból. A lépéssort addig folytassuk amíg a munkák el nem fogynak.

Megjegyzés: Az ütemezési vektorból kiemelt munkák végrehajtási sorrendje tetszőleges lehet, mert azok mind késnek! A feladat az ilyen munkák számának minimalizálása volt.

A Moor-algoritmus alkalmazása

Oldjuk meg az $1|d_i|\sum U_i$ ütemezési feladatot!

Alapadatok:

i	p_i	d_i
1	5	6
2	3	8
3	8	20
4	2	4

Megoldás: Rendezzük a munkákat EDD sorrendbe, vagyis a határidők alapján nemcsökkenő sorrendbe:

Pozíció	1	2	3	4
i	4	1	2	3

A Moor-algoritmus alkalmazása

Tegyük be a J_4 munkát az ütemezési vektorba: $s = \{ 4 \}$
Ekkor $t = C_4 = 2$. Nincs késés($d_4 = 4$)!

Tegyük be a J_1 munkát az ütemezési vektorba: $s = \{ 4, 1 \}$
Ekkor $t = C_1 = 7$. Mivel $d_1 = 6$, így a munka késik!

Válasszuk ki a beütemezettek közül a legnagyobb műveleti idejű munkát:
Ez a J_1 lesz. Vegyük ki az ütemtervből és tegyük félre: $s = \{ 4 \}$ és $V = \{ 1 \}$

Tegyük be a soron következő a J_2 munkát az ütemtervbe: $s = \{ 4, 2 \}$ és $V = \{ 1 \}$
Ekkor $t = C_2 = 2 + 3 = 5$. Mivel $d_2 = 8$, így nincs késés!

Tegyük be az utolsó J_3 munkát az ütemtervbe: $s = \{ 4, 2, 3 \}$ és $V = \{ 1 \}$
Ekkor $t = C_3 = 5 + 8 = 13$. Mivel $d_3 = 20$, így nincs késés!

A Moor-algoritmus alkalmazása

A végeredményt az S (nem késő munkák) halmazának és a V (késő munkák) halmazának egyesítése jelenti:

Pozíció	1	2	3	4
i	4	2	3	1

Foglaljuk táblázatba az eredményeket az ütemezési sorrend szerint!
Használjuk a korábban megtanult definíciókat!

Késés (Lateness): $L_i = C_i - d_i$

Csúszás (Tardiness): $T_i = \max\{0, L_i\}$

Egységnyi büntetés (Unit penalty): $U_i = \begin{cases} 1, & \text{ha } C_i > d_i \\ 0, & \text{egyébként} \end{cases}$

A Moor-algoritmus alkalmazása

Az eredmények táblázatba foglalása az ütemezési sorrend szerint:

i	p_i	d_i	C_i	L_i	T_i	U_i
4	2	4	2	-2	0	0
2	3	8	5	-3	0	0
3	8	20	13	-7	0	0
1	5	6	18	12	12	1

A célfüggvény értéke (a késő munkák száma):

$$\gamma = \sum U_i = \sum_{i=1}^{n} (U_i))$$

$$= U_1 + U_2 + U_3 + U_4 = 1+0+0+0=1$$

A Lawler-algoritmus alkalmazása

A feladat formális leírása: $1|\text{prec}|f_{max}$

Ahol $f_{max}=max_{i=1}^{n}\{f_i(C_i)\}$ és f_i monoton növekvő (reguláris) függvénye a C_i értéknek.

A feladat értelmezése: Egyetlen erőforráson adott számú munkát kell elvégezni. A tanult alapértelmezett végrehajtási jellemzők vannak érvényben, továbbá a munkák között sorrendi előírások lehetnek.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely megfelel a sorrendi kötöttségeknek és a választott f_i célfüggvény értéke minimális értéket vesz fel.

Az f_i szimbólum jelölheti többek között például az L_i késést vagy a T_i csúszást.

A Lawler-algoritmus alkalmazása

Az algoritmus bemutatásakor használt Jelölések:

J a munkák halmaza: $J = \{1, 2, \ldots, n\}$,

$i, j, j^* \in J$ az elvégzendő munkák szimbólumai,

$S \subseteq J$ a már beütemezett munkák halmaza,

$U = J \setminus S$ a még be nem ütemezett munkák halmaza (az S komplementere).

$V \subseteq U$ azoknak a még be nem ütemezett munkáknak a halmaza, amelyeknek az aktuális időpontban nincs be nem ütemezett rákövetkező munkájuk a precedencia gráfban.

A Lawler-algoritmus alkalmazása

A feladat megoldását végző algoritmus leírása:

1. lépés: Inicializálás.

Legyen kezdetben $S = \{\ \}$ üres halmaz, minden munka ütemezetlen $U = \{1, 2, \ldots, n\}$, és V tartalmazza azon munkákat, amelyekre nincs rákövetkező munka a precedencia gráfban.

2. lépés: Kiválasztás.

Válasszuk ki azt a $j^* \in V$ munkát, amelyre teljesül az alábbi feltétel:

$$f_{j^*}(\Sigma_{j\in U} p_j) = min_{i\in V}(f_i(\Sigma_{j\in U} p_j))$$

A Lawler-algoritmus alkalmazása

3. lépés: Aktualizálás.

Tegyük át a j^* kiválasztott munkát az U halmazból az S ütemtervbe a már beütemezett munkák elé és frissítsük a V halmaz tartalmát.

4. lépés: Feltételvizsgálat.

Ha az U halmaz üres akkor készen vagyunk, egyébként folytassuk a 2. lépéssel.

A bemutatott algoritmus az $1|\,\mathrm{prec}|\,f_{max}$ feladat optimális megoldását adja.

A Lawler-algoritmus alkalmazása

Az algoritmus értelmezése:

Az ütemterv építését hátulról előre haladva végezzük.

A munkákat egyesével rakjuk be az ütemezési vektorba a végéről (jobbról) az eleje felé (balra) haladva. A kiválasztás során az aktuálisan beütemezhető munkák közül választunk. Az a munka tekinthető kiválaszthatónak, amelyiknek nincs beütemezetlen rákövetkező munkája. A jelöltek közül azt a munkát választjuk ki, amelyiket a még beütemezetlen munkák műveleti idejének összegével behelyettesítve az fi függvénybe a legkisebb értéket kapjuk. A kiválasztott munkát betesszük az ütemezési vektorba közvetlenül a már beütemezettek elé, és kivesszük az ütemezetlen munkák közül. Ezt követően frissítjük a beütemezhető munkák halmazát (a kiválasztott munka beütemezése következtében további munkák válhatnak ütemezhetővé azáltal, hogy a precedencia gráfban a rákövetkező munkájuk most került éppen beütemezésre).

A Lawler-algoritmus alkalmazása

Az általános formában megfogalmazott feladat formális leírása:

$$1|\,\mathrm{prec}|\,f_{max}$$

ahol

$$f_{max} = max_{i=1}^{n}\{f_i(C_i)\}$$

és f_i monoton növekvő (reguláris) függvénye a C_i értéknek.

A Lawler-algoritmus alkalmazása

Az általánosított feladat $(1|\text{prec}|f_{max})$egy speciális esete a következő:

$$1|\ \text{prec},\ d_i|\ L_{max}$$

ahol $L_{max} = max_{i=1}^{n}\{L_i(C_i)\}$

Mivel $f_{max} = L_{max}$ és $f_i(C_i) = L_i(C_i) = C_i - d_i$

Egyetlen erőforráson adott számú munkát kell elvégezni. A tanult alapértelmezett végrehajtási jellemzők vannak érvényben, továbbá a munkák között sorrendi előírások lehetnek.

Az ütemezés célja az, hogy válasszunk egy olyan végrehajtási sorrendet, amely megfelel a sorrendi kötöttségeknek és a legnagyobb mértékű határidő túllépés minimális legyen.

A Lawler-algoritmus alkalmazása

Megjegyzés:

Az $1|\mathrm{prec}|f_{max}$ feladat speciális esete a korábban részletesen vizsgált $1||d_i|L_{max}$ feladat is. A sorrendi korlátozások nélküli esetben a Lawler algoritmus az EDD szabályra egyszerűsödik.

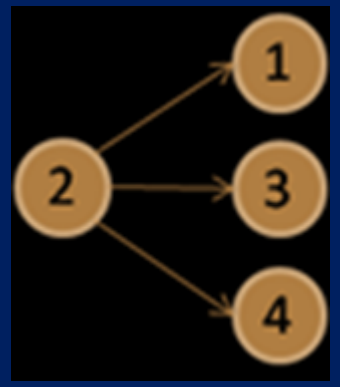
A Lawler-algoritmus alkalmazása

Feladat: Oldjuk meg az $1|\,\mathrm{prec},\, d_i|\, L_{max}$ ütemezési feladatot!

Alapadatok:

i	p_i	d_i
1	5	15
2	3	18
3	8	14
4	2	18

Az előírt megelőzési relációkat ábrázoló precedencia gráf:

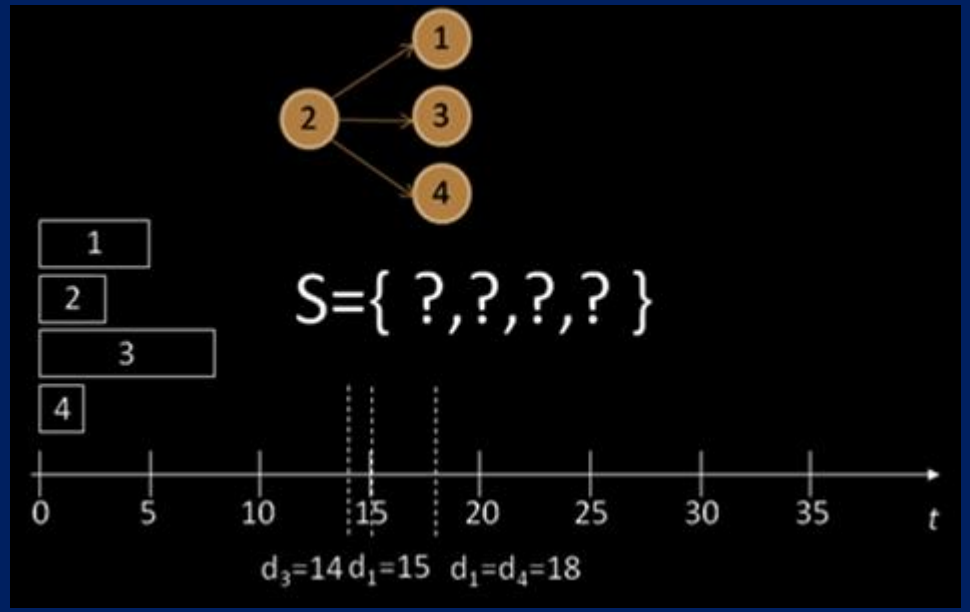


A Lawler-algoritmus alkalmazása

Megoldás:

0. induló állapot

Inicializálás: S = { }, U = { 1, 2, 3, 4 }, V = { 1, 3, 4 }



A Lawler-algoritmus alkalmazása

Válasszuk ki a $j^* \in V$ munkát figyelembe véve, hogy most $f_i = L_i$.

Először számítsuk ki az aktuális befejezési időpontot:

$\sum_{j \in U} p_j = p_1 + p_2 + p_3 + p_4 = 5+3+8+2 = 18$

A kapott eredményt helyettesítsük be az Li függvénybe:

$L_{j^*}(\sum_{j \in U} p_j) = L_{j^*}(18) = min_{i \in V} (L_i(18))$

$= \min(L_1(18), L_3(18), L_4(18))$

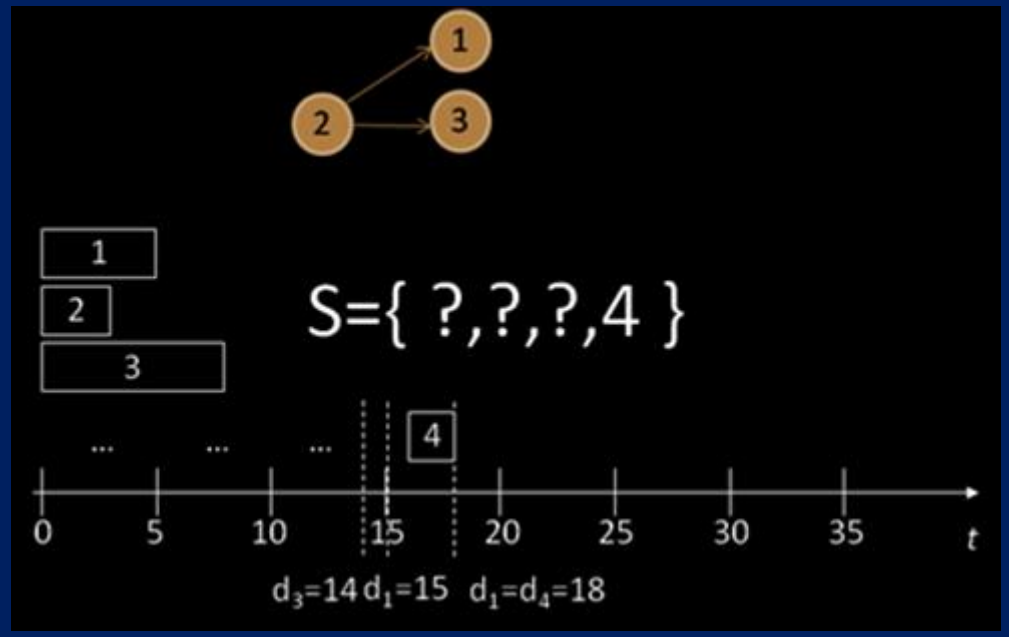
$= \min(18-15, 18-14, 18-18) = L_4 = 0$

$j^* = 4$

A Lawler-algoritmus alkalmazása

Aktualizálás: Tegyük át a j^* kiválasztott munkát az U halmazból az S ütemtervbe a már beütemezett munkák elé és frissítsük a V halmaz tartalmát.

$S = \{ 4 \}, U = \{ 1, 2, 3 \}, V = \{ 1, 3 \}$



A Lawler-algoritmus alkalmazása

Feltételvizsgálat: Az U halmaz nem üres! Folytassuk a következő elem kiválasztásával!

$$\sum_{j \in U} p_j = p_1 + p_2 + p_3 = 5+3+8=16$$

$$L_{j^*}\left(\sum_{j \in U} p_j\right) = L_{j^*}(16) = min_{i \in V}\left(L_i(16)\right)$$

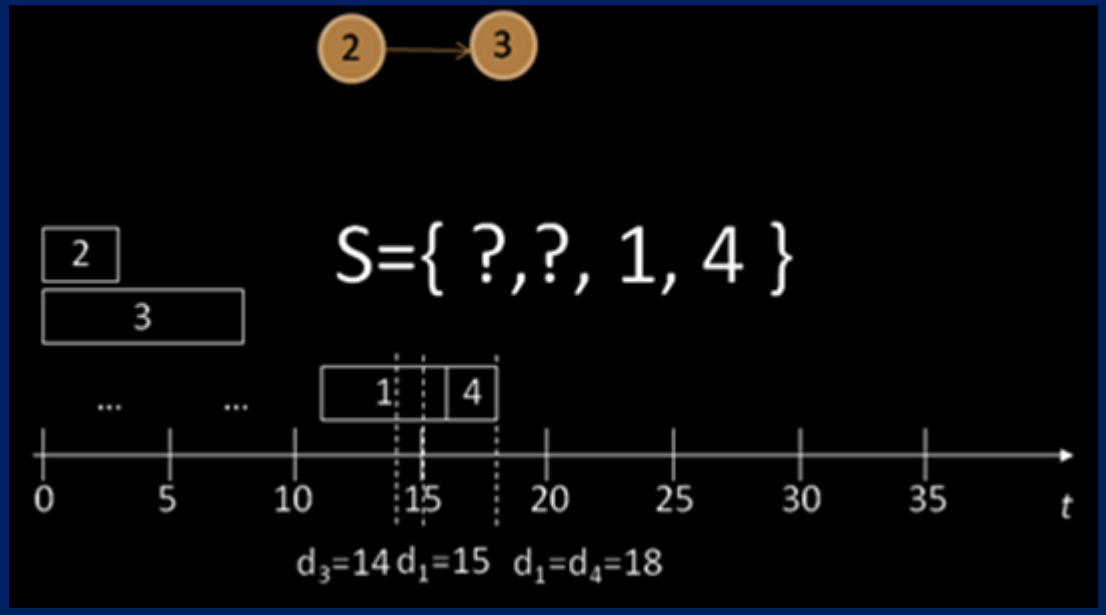
$$= \min(L_1(16), L_3(16)) = \min(16-15, 16-14) = L_1 = 1$$

$$j^* = 1$$

A Lawler-algoritmus alkalmazása

Aktualizálás: a kiválasztott munkát (1) behelyezzük a már beütemezett munka (4) elé.

$S = \{ 1, 4 \}, U = \{2, 3\}, V = \{ 3 \}$



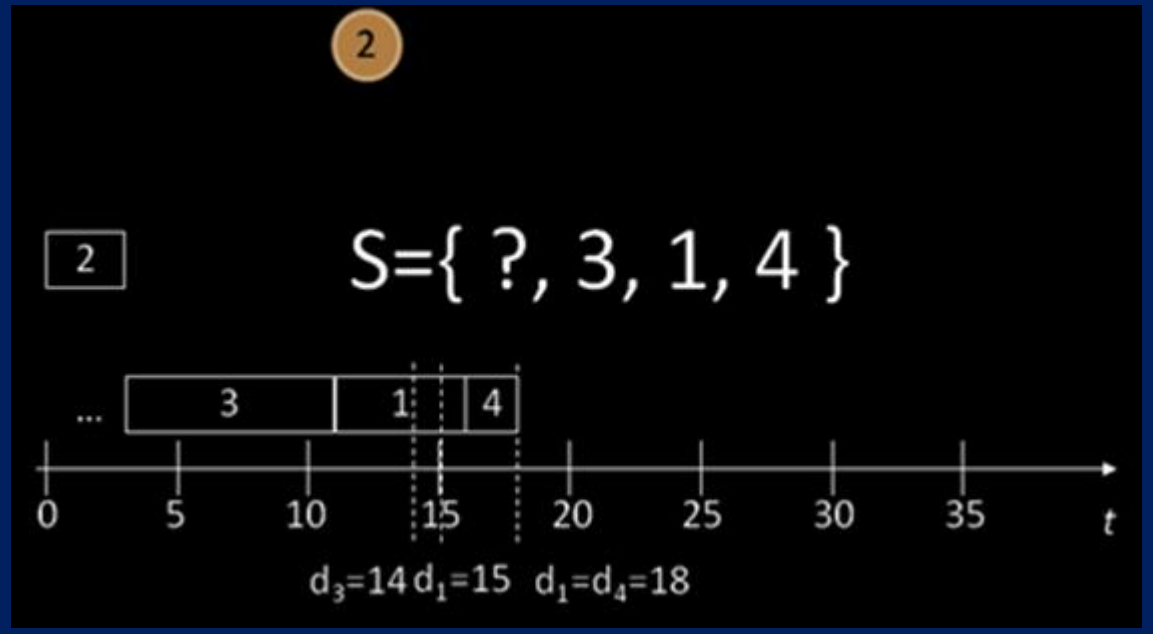
A Lawler-algoritmus alkalmazása

Feltételvizsgálat: Az U halmaz nem üres! Folytassuk a következő elem kiválasztásával!

Ebben az esetben nincs választás mert csak egy jelölt munka van: $j^* = 3$

Aktualizálás: a kiválasztott munkát (3) behelyezzük a már beütemezett munkák (1 és 4) elé.

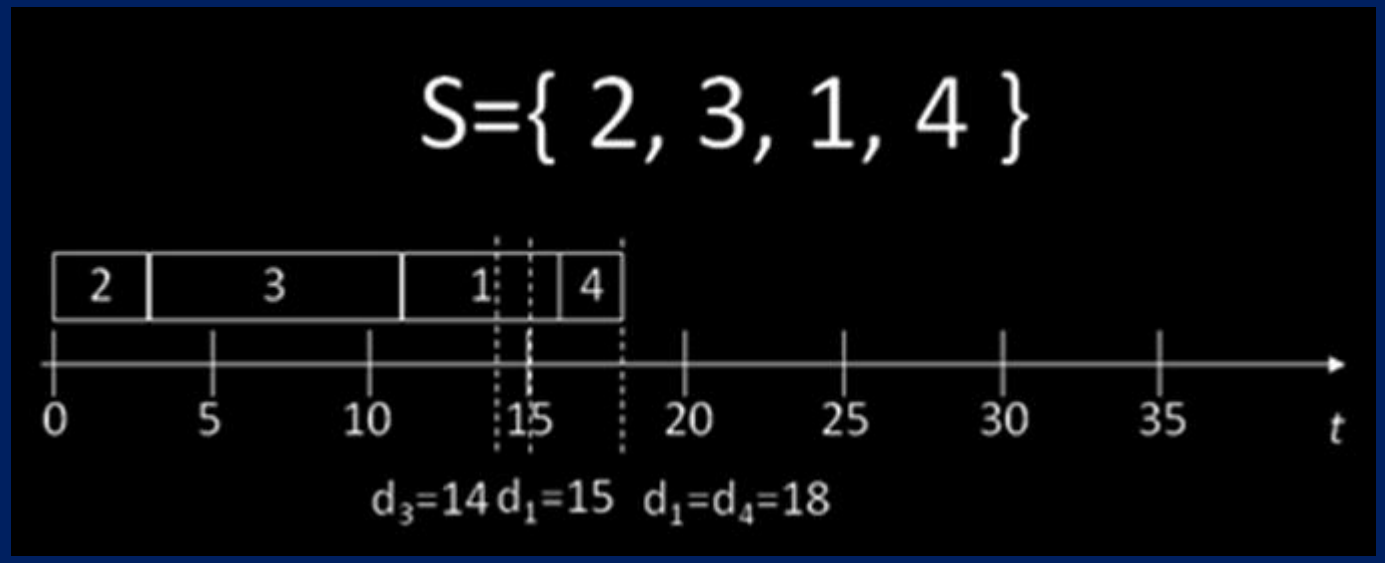
$S = \{ 3, 1, 4 \}, U = \{ 2 \}, V = \{ 2 \}$



A Lawler-algoritmus alkalmazása

Most már a 2. munka is kiválaszthatóvá vált: $j^* = 2$

$S = \{ 2, 3, 1, 4 \}, U = \{ \}, V = \{ \}$



Az U halmaz üres! Készen vagyunk! Minden munkát beütemeztünk.

A Lawler-algoritmus alkalmazása

Foglaljuk táblázatba az eredményeket az ütemezési sorrend szerint!

i	p_i	d_i	C_i	L_i	T_i
2	3	18	3	-15	0
3	8	14	8	-6	0
1	5	15	16	1	1
4	2	18	18	0	0

A célfüggvény értéke (a maximális késés): $\gamma = L_{max} = max_{i=1}^{n}\{L_i\}$

$$= \max\{L_1, L_2, L_3, L_4\} = \max\{1, -15, -6, 0\} = 1$$

A maximális csúszás: $T_{max} = max_{i=1}^{n}\{T_i\}$

$$= \max\{T_1, T_2, T_3, T_4\} = \max\{1, 0, 0, 0\} = 1$$

Köszönöm a figyelmet!

Az előadásvázlat elérhető az alábbi webcímen:

http://ait.iit.uni-miskolc.hu/~kulcsar/servo7.htm