Molnár S. Verona

A SORBARENDEZÉSI PROBLÉMÁK NÉHÁNY MEGOLDÁSI MÓDSZERE ÉS ÖSSZEHASONLÍTÁSA

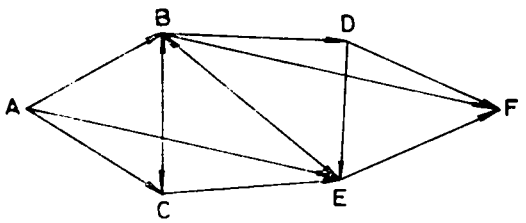
Bevezető

A mindennapi életben, bizonyos szervezési problémák megoldásakor gyakran jelentkeznek sorbarendezési, illetve ütemezési feladatok. Ezekre a feladatokra jellemző, hogy olyan tevékenységek szerepelnek bennük, amelyek különböző sorrendben valósulhatnak meg. Az ilyen probléma pl. a közgazdaságban ismert termelési műveletek sorbarendezése minimális gépterhelés, minimális átállítási költség, vagy legrövidebb várakozási idő elérésére. A gyakorlatban felmerülő sorbarendezési feladatok megoldásánál jelentkezik az a probléma, hogy a különböző módszereket alkalmazó szakemberek nem ismerik eléggé az egyes módszerek hatókörét, s ezeknek viszonylagos értékeit. Tehát mondhatjuk, hogy a gyakorlat tette fel azt a kérdést, amelyre munkánkban válaszolni szeretnénk. Ebből adódóan munkánk célja nemcsak az egyes módszerek általánosítása és leírása, hanem viszonylagos értékelésük, összehasonlításuk, rendszerezésük, illetve klasszifikációjuk.

Munkánk első részében a gyakorlatban jelentkező sorbarendezési problémák megoldásának néhány módszerét mutatjuk be. E módszerek közös tulajdonsága, hogy alkalmazásuk igen egyszerű, nem igényel különösebb matematikai-szakmai tudást. A munka második része tartalmazza az összehasonlító kritériumokat, az összehasonlító táblázatot, illetve az összehasonlító értékelést. Az összehasonlító táblázatban az algoritmusokat bizonyos előre megállapított kritériumok szerint jellemeztük, s így ki tudtuk emelni a módszereknek egyes előnyeit másokkal szemben. A gazdasági és műszaki szakemberek feladata ezek után a gyakorlati probléma meghatározására és leírására szorítkozik. Ugyanis a megfelelő sorbarendezési algoritmus kiválasztása már nem bonyolult feladat.

1. Foulkes algoritmus

Az algoritmus alkalmazásának menetét egy feladaton szeretnénk bemutatni. Hat tevékenységet kell sorbarendezni, ezek az A, B, C, D, E és F tevékenységek. A probléma elképzelhető, mint valamilyen műveletek sorozata egy termék összeállításakor (szerelésekor), ahol ismertek a tevékenységek közötti összefüggések, vagyis az, hogy mely tevékenység melyiket előzi meg, s melyik után következik. Az ismert összefüggések alapján a tevékenységeket gráf formájában ábrázoljuk, mégpedig úgy, hogy a hat tevékenység jelenti a gráf csúcsait, az összefüggéseket pedig a gráf ívei jelzik. Az íven lévő nyíl az összefüggés irányát mutatja.



Az egyes ívekben levő kettős nyíl azt jelenti, hogy a két csúcs (tevékenység) sorrendje nem lényeges, vagyis, hogy pl. a B lebonyolódhat a C előtt, de következhet a C után is.

Feladatunk az, hogy megkeressük, ha létezik, azt a hamiltoni utat, amely a belépő és kilépő, esetünkben az A és F között van. Ugyanis a hamiltoni út csak egyszer halad át az egyes csúcsokon s ugyanakkor kielégíti az adott összefüggéseket. A feladat értelmében az A, B, C, D, E és F tevékenységeket bizonyos sorrendben kell elvégezni, de egyiket sem szabad kihagyni. A gráfban annyi hamiltoni út van, ahányféleképp sorba rakhatók a csúcsok. Az össz lehetőségek száma 6 csúcs esetén 6! = 720. Szerencsére arra nincs szükség, hogy az össz lehetőséget sorra kipróbáljuk, azért, hogy ezáltal megállapítsuk, mely kombinációk elégítik ki az össz feltételeket és követelményeket. Ugyanis a valóságban az össz sorbarendezési lehetőségek jó része kiesik, mert azok eleve nem elégítik ki az össz követelményeket. A hamiltoni út hossza n-ed rendű gráfban egyenlő (n—1)-el, ahol n a csúcsok száma. Esetünkben a gráf 6-od rendű, mert 6 csúcs van, a hamiltoni út hossza pedig 5. Az a teendő tehát, hogy megkeressük a gráfban levő 5 hosszúságú utat, s ezek közül kiválasszuk azt, vagy azokat, amelyek minden feltételnek eleget tesznek.

A probléma megoldásához a Foulkes algoritmust alkalmazzuk.[1] A Foulkes módszer lehetővé teszi, hogy megkeressük a gráfban szereplő 1, 2, ..., n—1 hosszúságú utakat, mégpedig úgy, hogy a gráfot reprezentáló M mátrixot önmagával annyiszor szorozzuk, amilyen hosszúságú utat keresünk. A szorzás elvégzésekor azonban, a Boole-féle szorzást kell alkalmaznunk. Ha a csúcsok száma n, és ha meghatározzuk a

$$([1] \cdot [M])^{n-1}$$

2. *ábra* mátrixot, akkor megkapjuk az (n—1) vagy attól rövidebb hosszúságú utakat. Mint tudjuk, a Boole-féle összeadásnál a következő szabály érvényes: $1+0=1$, $0+1=1$, $1+1=1$, $0+0=0$, míg a szorzásnál: $1 \cdot 0=0$, $0 \cdot 1=1$, $1 \cdot 1=1$, $0 \cdot 0=0$.

Az összefüggéseket jelző gráfnak most felírjuk a mátrix reprezentációját. A kapott M mátrixnak annyi sora és oszlopa van, amennyi csúcsa a gráfnak, illetve ahány tevékenység vagy művelet. Az M mátrixnak csak azon sorába és oszlopába írunk egyet, amely élek között létezik irányított út. A mátrix fő átlóján szintén egyeseket írunk.

$$M = \begin{array}{c|cccccc} & A & B & C & D & E & F \\ A & 1 & 1 & 1 & 0 & 1 & 0 \\ B & 0 & 1 & 1 & 1 & 1 & 1 \\ C & 0 & 1 & 1 & 0 & 1 & 0 \\ D & 0 & 0 & 0 & 1 & 1 & 1 \\ E & 0 & 1 & 0 & 0 & 1 & 1 \\ F & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Az M mátrixnak önmagával történő szorzását a következőképpen végezzük el. Minden egyes új elem kiszámításához az elemhez tartozó sort és oszlopot összeszorozzuk a Boole-féle szorzatot alkalmazva. Pl. vegyük a B sorát és C oszlopát.

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = 0 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 0 = 0 + 1 + 1 + 0 + 0 + 0 = 2$$

Az egész mátrix szorzása így néz ki:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 3 & 1 & 4 & 2 \\ 0 & 1 & 2 & 2 & 4 & 4 \\ 0 & 3 & 1 & 2 & 3 & 2 \\ 0 & 1 & 0 & 1 & 2 & 3 \\ 0 & 2 & 1 & 1 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

A most kapott szorzatokon el kell végezni a helyettesítéseket, mégpedig úgy, hogy a főátlóra 1-eseket írunk, az 1-nél nagyobb számokat 1-gyel helyettesítjük, a 0-t változatlanul hagyjuk. A helyettesítések után a következő mátrixot kapjuk, melyet M^2-vel jelölünk:

$M^{(2)} =$

	A	B	C	D	E	F
A	1	1	1	(1)	1	(1)
B	0	1	1	1	1	1
C	0	1	1	(1)	1	(1)
D	0	(1)	0	1	1	1
E	0	1	(1)	(1)	1	1
F	0	0	0	0	0	1

Zárójelbe tettük azokat az 1-eseket, amelyek az eredeti **M** mátrixban nem szerepeltek. Ez mutatja meg, hogy mely csúcsok között van 2 hosszúságú út, de nincs 1 hosszúságú. Az M^2 mátrixot újból önmagával szorozzuk, mégpedig azért, hogy megkapjuk, mely csúcsokat köt össze 4, vagy 4-nél rövidebb hosszúságú út. Az M^2 önmagával történő szorzata az előző szabályok szerint történik (a Boole szorzás alkalmazásával). Az eredmény a helyettesítés után a következő:

$M^{(4)} =$

	A	B	C	D	E	F
A	1	1	1	1	1	1
B	0	1	1	1	1	1
C	0	1	1	1	1	1
D	0	1	(1)	1	1	1
E	0	1	1	1	1	1
F	0	0	0	0	0	1

A mátrixban még szerepel olyan új egyes (zárójelben megjelölve) amely az előzőekben nem szerepelt. Ez azt jelenti, hogy a sorban szereplő csúcstól az oszlopban szereplő csúcsig (tehát D-től C-ig) található 4, vagy ennél rövidebb út. Minthogy mi az 5 hosszúságú utakat keressük, ezért folytassuk a mátrixnak önmagával történő szorzását. Most megszorozzuk az M^4 mátrixot az **M** mátrixszal, hogy megkapjuk az M^5-öt, ami megmutatja, mely csúcsokat köt össze 5 és ennél rövidebb hosszúságú út. A szorzás és helyettesítés után a következő mátrixot kapjuk:

$M^{(5)} =$

1	1	1	1	1	1
0	1	1	1	1	1
0	1	1	1	1	1
0	1	1	1	1	1
0	1	1	1	1	1
0	0	0	0	0	1

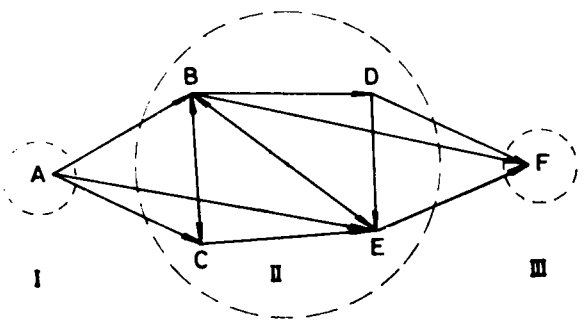
A kapott mátrix nem tartalmaz új 1-est, ugyanaz, mint az előző M^4 mátrix. Itt abba kell hagyni a szorzást. Meg kell keresni a halmaz hamiltoni útját. Ez úgy történik, hogy a mátrixnak azon sorát töröljük, amelyben csupa 1-esek vannak, s töröljük a csúcshoz tartozó oszlopot

is. Ez annyit jelent, hogy az A csúcs mindegyik rákövetkező tevékenységet megelőz, ez képezi az I. osztályt. Az A sorának és oszlopának törlése után a következő mátrixot kapjuk:

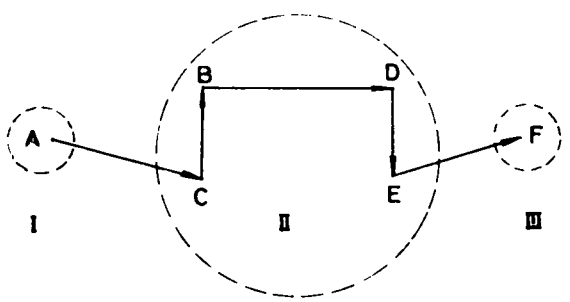
	B	C	D	E	F
B	1	1	1	1	1
C	1	1	1	1	1
D	1	1	1	1	1
E	1	1	1	1	1
F	0	0	0	0	1

Itt szintén töröljük az 1-eseket tartalmazó sorokat és a hozzájuk tartozó oszlopokat. Ezek a B, C, D és E sorok. Ezek alkotják a II. osztályt. A megmaradt táblázatban nincsenek 0-k. Az F képezi a III. osztályt.

Az osztályokon belül meg kell keresni a hamiltoni utakat. Az az M^5 mátrix sorainak és oszlopainak átrendezésével történik, amely e munka keretén belül nincs letisztázva. Alkalmasabbnak látszik az osztályokra bontás elvégzése magán a gráfon. Az osztályok ugyanis össze vannak kötve. A csoportok között most meg kell keresni azokat az utakat, amelyek összekötik az osztályok hamiltoni útjait. Az I. osztályt csak az A képezi, a II. osztályban a C, B, D és E képezi a hamiltoni utat, a III. osztályban csak az F van.



Ez azt jelenti, hogy a gráfról most leolvasható, hogy a feltételeket csak a következő sorrendben rakott tevékenységek elégítik ki: A, C, B, D, E és F.



A gyakorlatban sokszor jelentkezik olyan probléma, amelynél valamilyen gépen bizonyos átállítási költséggel vagy várakozással több terméket lehet előállítani. Ilyen esetben a termékek megmunkálásának olyan sorrendjét keressük, amely a legkevesebb átállítási költséggel, vagy várakozással jár. Ilyen esetben szintén alkalmazható Foulkes módszere.

Induljunk ki egy példából. Ismert 6 terméknek egy adott gépen történő átállítási költsége, egy mátrix alakjában. Meg kell keresni a termékek feldolgozásának sorrendjét, úgy, hogy az átállítási költség a legkisebb legyen. A következő mátrix a c_{ij} (i, j=1, 2, . . ., 6) átállítási költségeket tartalmazza:

	A	B	C	D	E	F
A	0	2	3	3	1	4
B	3	0	4	1	2	1
C	1	2	0	2	2	4
D	3	1	2	0	3	2
E	4	5	3	1	0	1
F	2	1	2	3	4	0

Ebből a táblázatból egy új táblázatot kell szerkeszteni, amelyre a következő szabályt alkalmazzuk:

a) ha $c_{ij} = c_{ji}$, az egyik költség helyébe 0-t, a másik helyébe pedig 1-et írunk,

b) ha $c_{ij} > c_{ji}$, a c_{ji} helyébe írunk 1-est, c_{ij} helyébe 0-t,

c) ha $c_{ij} < c_{ji}$, akkor c_{ij} helyébe írunk 1-est, c_{ji} helyébe pedig nullát,

d) a fő átlóra 1-eseket írunk.

A szabály alkalmazása után a következő mátrixot kapjuk:

	A	B	C	D	E	F
A	1	1	0	1	1	0
B	0	1	0	0	1	0
C	1	1	1	1	1	0
D	0	1	0	1	0	1
E	0	0	0	1	1	1
F	1	1	1	0	0	1

Ezt a mátrixot a továbbiakban ugyanúgy kezeljük, mint az előző esetben a gráfot reprezentáló mátrixot, tehát a Foulkes módszer szerint megkeressük a gráfon levő osztályokat.

Most lépésekbe foglaljuk a sorbarendezési problémára alkalmazható Foulkes módszert:

1. lépés: A probléma tevékenységeit és azoknak összefüggéseit gráf alakjában ábrázoljuk.

2. lépés: A gráfot egy olyan M mátrix segítségével írjuk fel, amelynek csak 0 és 1 elemei vannak, valamint annyi sora és oszlopa,

amennyi csúcsa a gráfnak. A mátrixnak csak azon eleme lesz 1, amely két csúcs közötti kapcsolatot, illetve út létezését jelzi.

3. lépés: A bizonyos M mátrixot megszorozzuk önmagával úgy, hogy a Boole-féle szorzatot alkalmazzuk.

4. lépés: Az 1 és 1-nél nagyobb elemeket 1-gyel helyettesítjük s így megkapjuk az M^2 mátrixot.

5. lépés: A mátrix önmagával való szorzását legalább (n—1)-szer végezzük el. Ahol az n a csúcsok száma, alkalmazva a 3. és 4. lépést. A mátrix önmagával történő szorzását akkor hagyjuk abba, amikor a szorzás után ugyanazt a mátrixot kapjuk, mint a szorzás előtt.

6. lépés: Meghatározzuk a gráf osztályait. Az osztályokra bontás a következőképpen történik: az I. osztályt azok a csúcsok alkotják, amelyeknek sorában csupa egyes található. A csúcshoz tartozó sort és oszlopot töröljük. A II. osztályba azon csúcsok tartoznak, amelyek sorában a megmaradt mátrixban csupa 1-esek vannak. A csúcshoz tartozó sort és oszlopot töröljük. A következő osztályokat is az előző módon határozzuk meg. A műveletet akkor fejezzük be, amikor a megmaradt mátrix csupa 1-eseket tartalmaz. A sorokhoz tartozó csúcsok alkotják az utolsó osztályt.

7. lépés: Az osztályokon belül megkeressük a hamiltoni utakat. Azt az utat, amely minden csúcson csak egyszer halad keresztül, az osztályonkénti hamiltoni utak összegezése után kapjuk. A kapott hamiltoni út adja a tevékenységek keresett sorrendjét.

8. lépés: A hamiltoni út és az eredeti mátrix alapján kiszámítható az össz átállítási költség nagysága.

2. *Oszlopvektoros eljárás*

Az előző pontban ismertetett sorbarendezési problémát oszlopvektoros eljárással is megoldhatjuk.[2] A tevékenységek összefüggéseit itt is gráf alakjában ábrázoljuk, amelynek felírjuk a mátrix reprezentációját az előzőekben ismertetett módon. A kapott mátrixon végezzük el a számításokat. Az eredetihez viszonyítva az eljárást bizonyos szempontból módosítottuk. A módosítást azért vezettük be, mert megítélésünk szerint nem alkalmazható minden gyakorlati példára. A módosítás után véleményünk szerint ez lehetővé vált. Mindenesetre ennek bővebb és részletesebb igazolása további kutatás célja marad. A módosításokat a megfelelő helyen fel is tüntettük.

A gráf mátrixreprezentációjának oszlopvektorait $\overline{V_A}$, $\overline{V_B}$, $\overline{V_C}$ $\overline{V_D}$, $\overline{V_E}$ és $\overline{V_F}$-fel jelöljük. A mátrix fő átlójára az eredeti oszlopvektoros eljárástól eltérően 1-eseket írunk.

	$\vec{V}_A$	$\vec{V}_B$	$\vec{V}_C$	$\vec{V}_D$	$\vec{V}_E$	$\vec{V}_F$	$\vec{V}_0$	$\vec{V}_1$	$\vec{V}_2$	$\vec{V}_3$
A	1	1	1	0	1	0	4	4	3	(1)
B	0	1	1	1	1	1	5	4	(2)	0
C	0	1	1	0	1	0	3	3	(2)	0
D	0	0	0	1	1	1	3	(2)	0	0
E	0	1	0	0	1	1	3	(2)	0	0
F	0	0	0	0	0	1	(1)	0	0	0
					szint:		0	1	2	3
					csúcsok:		F	D E	C B	A

Most meg kell határozni a $\vec{V}_0$ vektor értékét, mégpedig a következő összeadással:

$$\vec{V}_0 = \vec{V}_A + \vec{V}_B + \vec{V}_C + \vec{V}_D + \vec{V}_E + \vec{V}_F$$

Az összeadás eredményét a mátrix folytatásában levő táblázatban adjuk meg. A kapott $\vec{V}_0$ vektorban megkeressük a legkisebb számot. Esetünkben ez az 1-es, amely az F sorában van. Tehát az F tartozik a 0 szinthez, ami annyit jelent, hogy nincs leszármazottja, tehát utolsó, befejező tevékenység. A táblázatban a megfelelő sorban bekarikáztuk az 1-est. Töröljük a csúcshoz tartozó sort és oszlopot, majd az 1. szint megkeresésére kiszámítjuk a $\vec{V}_1$-et, úgy, hogy összeadjuk a megmaradt oszlopvektorokat:

$$\vec{V}_1 = \vec{V}_A + \vec{V}_B + \vec{V}_C + \vec{V}_D + \vec{V}_E$$

Az eredeti oszlopvektoros eljárás szerint a $\vec{V}_1$ vektort a következőképpen kaphatjuk meg:

$$\vec{V}_1 = \vec{V}_0 - \vec{V}_F$$

Ez a különbség egyébként megegyezik az előzőekben kiszámított összeggel.

Az összeadás eredményét feltüntetjük a táblázatban, s megkeressük a legkisebb számokat. Ezek a D és E oszlopban vannak. Az 1. szintet a D és E csúcsok alkotják. Töröljük a hozzájuk tartozó sort és oszlopot, majd meghatározzuk a $\vec{V}_2$ vektort, úgy mint a megmaradt oszlopvektorok összegét:

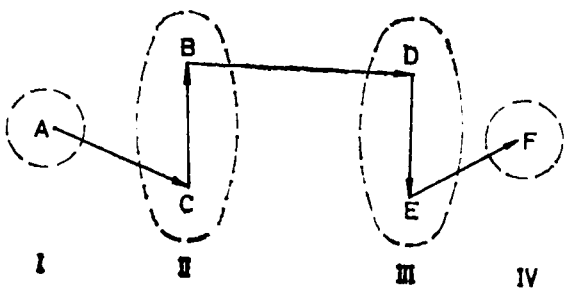
$$\vec{V}_2 = \vec{V}_A + \vec{V}_B + \vec{V}_C$$

vagyis az eredeti eljárás alapján a $\vec{V}_1$ és az 1. szinthez tartozó oszlopvektorok különbségét. A legkisebb számok a B és C sorában vannak,

ezért ezek a csúcsok alkotják a 2. szintet. A csúcsokhoz tartozó sorokat és oszlopokat töröljük, s meghatározzuk a 3. szintet:

$$\overrightarrow{V_3} = \overrightarrow{V_A}$$

A kapott szinteket fordított sorrendben számozva megkapjuk az I., II., a III. és a IV. osztályokat illetve szinteket. Ezeket most a gráfon is feltüntetjük, s meg is határozzuk a hamiltoni utat.



Mint látjuk az osztályokra bontás nem adott azonos eredményt a Foulkes eljárással kapott osztályokra bontással. Ez viszont nem akadálya annak, hogy azonos hamiltoni utakat állapítsunk meg. Vagyis a tevékenységekre azonos sorbarendezést kaptunk. A termékeket a következő sorrendben kell feldolgozni: A, C, B, D, E, F. A módszert most lépésekbe foglaljuk:

1. lépés: A probléma tevékenységeit és azoknak összefüggéseit gráf alakjában ábrázoljuk.

2. lépés: A gráfot egy olyan M mátrixszal reprezentáljuk, amelynek csak 0 és 1 elemei vannak, valamint annyi sora és oszlopa, amennyi csúcsa a gráfnak. A mátrixnak azon eleme lesz 1, amely két csúcs közötti kapcsolat, illetve út létezését jelzi.

3. lépés: Az M mátrix oszlopvektoraiból megalkotjuk a $\overrightarrow{V_0}$, $\overrightarrow{V_1}$, $\overrightarrow{V_2}$, . . ., $\overrightarrow{V_n}$ vektorokat, amelyeket az oszlopvektorok összegéből kapunk meg.

4. lépés: A $\overrightarrow{V_0}$ oszlopvektor legkisebb eleméhez tartozó csúcs alkotja a 0 szintet.

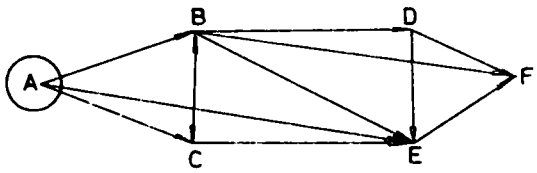
5. lépés: A 4. lépésben meghatározott csúcs sorát és oszlopát töröljük. A megmaradt mátrix alapján, áttérve a 3. és 4. lépésre, meghatározzuk a következő szinteket.

6. lépés: A $\overrightarrow{V_1}$, $\overrightarrow{V_2}$, $\overrightarrow{V_3}$, . . ., $\overrightarrow{V_n}$ vektorok meghatározását addig végezzük, míg van szabad oszlopvektor. Ha nincs, akkor a szinteket fordított sorrendben számozva megkapjuk a gráf osztályait.

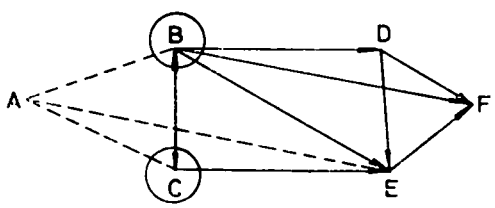
7. lépés: Az osztályokon belül megkeressük a hamiltoni utakat. Az egész gráfra vonatkozó hamiltoni utat az egyes osztályok hamiltoni útjainak összegeként kapjuk, amely megmutatja a tevékenységek keresett sorrendjét.

3. Grafikus eljárás

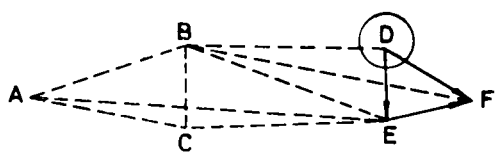
Az osztályokra bontás egy adott tevékenységi hálón (gráfon) közvetlenül is elvégezhető.[3] Néha igen kényelmes megoldás, de az a gráf elrendezésétől és tulajdonságától függ. Induljunk ki az előzőekben ismertetett problémából:



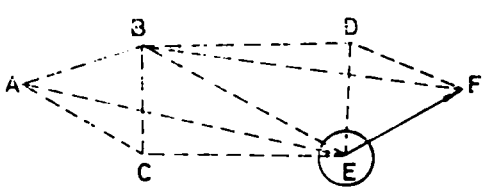
Nem kell mást tenni, mint lépésenként meghatározni az ős nélküli csúcsokat. Esetünkben ez a csúcs A. Ez alkotja az I. osztályt. Ezt a csúcsot, valamint a belőle kifutó éleket elhagyva egy új gráfot kapunk:



Itt is megkeressük az ős nélküli csúcsokat, ezek a B és C. Ezek alkotják a II. osztályt. Elhagyjuk a csúcsot és a hozzá tartozó íveket, s a következő gráfot kapjuk:

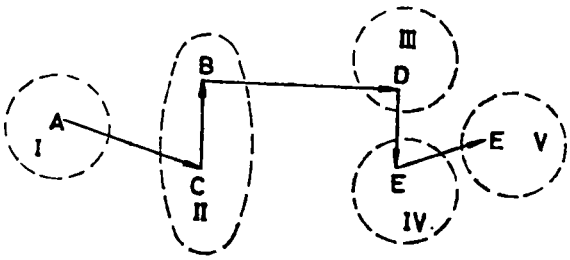


Az ős nélküli csúcs itt a D, ez alkotja a III. osztályt. A csúcs és a hozzá tartozó élek elhagyása után a következő gráfot kapjuk:



Itt az E az ős nélküli csúcs, s a IV. osztályt alkotja. A csúcs és élek elhagyásával csak az F csúcs marad, amelynek már nincs leszármazottja. Ez alkotja az V. osztályt.

Mint láttuk, az előzőekhez viszonyítva itt öt osztályt határoztunk meg. Ezek alapján könnyen megállapítható az a hamiltoni út, amely minden csúcson csak egyszer, de keresztülhalad. A hamiltoni út ugyanaz, mint amit az előző pontokban határoztunk meg:



Most határozzuk meg a módszer lépéseit:

1. lépés: A probléma tevékenységeit és azon összefüggéseit gráf alakjában ábrázoljuk.

2. lépés: Meghatározzuk az ős nélküli csúcsokat. Ezek alkotják az első, második stb. osztályokat.

3. lépés: A gráfban töröljük a 2. pontban kiválasztott ős nélküli csúcsokat és a hozzájuk tartozó össz íveket. Visszatérünk a 2. és 3. lépésre, s addig ismételjük, míg el nem fogynak az ős nélküli csúcsok, illetve míg olyan csúcsokhoz jutunk, melynek már nincsenek leszármazottjai.

4. lépés: A 2. lépésben meghatározott ős nélküli csúcsok alkotta osztályokban meghatározzuk a hamiltoni utakat. Az össz gráf hamiltoni útjai, az egyes hamiltoni utak összege lesz.

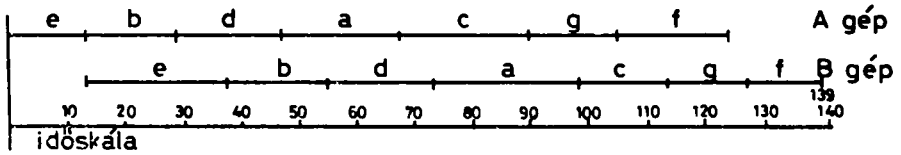
4. Johnson algoritmusa[4]

Johnson algoritmusa olyan problémák megoldására alkalmas, amelyeknél a feladatot kell két egymást követő munkahelyen, művelethez, vagy tevékenységhez hozzárendelni. Munkánkban a módszer matematikai indoklását nem tartottuk elsősorban feladatnak.

A módszert egy gyakorlati példán szeretnénk bemutatni. Legyen adott A és B gép, i darab alkatrész, melyeket a gépeken kell megmunkálni A_i és B_i idő alatt. A feladat az, hogy meghatározzuk, milyen sorrendben dolgozzuk fel a munkadarabokat, hogy a várakozási idő minimális legyen.

műveletek	idő (percekben)	
	A gépen	B gépen
a	20	25
b	15	17
c	22	15
d	18	18
e	13	24
f	19	13
g	15	14

A megoldás menete a következő: megvizsgáljuk a két művelethez szükséges idők táblázatát és kiválasztjuk a legkisebb értéket. Ha ez az érték az első gépen történő művelethez tartozik, akkor ezzel kezdünk, ha második gépen levő művelethez, akkor ezzel fejezzük a műveleteket. Esetünkben az f műveletet a B gépen lehet a legrövidebb idő alatt elvégezni, tehát f-fel végzünk. Az f sorát töröljük a táblázatból. Az e művelethez tartozó idő szintén a legrövidebb, 13 de az A gépen, tehát e-vel kezdünk, s töröljük sorát. A következő legkisebb érték a 14, ami a g műveletnek a B gépen való elvégzéséhez szükséges, tehát a g az f-et előzi meg. Ezen szabály szerint a következő sorrendet kapjuk: e, b, d, a, c, g, f. Az ábrán látható a műveletek elrendezése és tartama. Az össz időtartam 139 perc.



Jhonson módszerét a következő lépésekbe foglalhatjuk össze:

1. lépés: Megvizsgáljuk a két műveletre, illetve a két gépen szükséges időtartamokat, és az össz értékekből kiválasztjuk a legkisebbet.

2. lépés: Ha a legkisebb érték az első művelethez tartozik (ill. az első gépen történik), akkor ezzel kezdünk, ha viszont a második művelethez, akkor ezzel fejezzük a műveleteket.

3. lépés: Ezen művelethez tartozó sort töröljük, és újból áttérünk az 1., 2. és 3. lépésre. Mindezt addig folytatjuk, míg el nem fogynak a műveletek.

4. lépés: A kapott sorrend alapján könnyen ki lehet számolni azt a legrövidebb időt, amely elegendő az össz műveleteknek mindkét gépen való elvégzéséhez, illetve az átfutási időt.

A Johnson algoritmus kiterjeszthető olyan problémák megoldására, melyeknél 3 gépen, munkahelyen, kell elvégezni bizonyos műveleteket minden munkadarabon. Erre az esetre most nem térünk ki.

5. *Index módszer*[5]

Több gép közötti terhelés megosztására alkalmazható közelítő eljárás az index módszer is, amellyel elérhető a termékek gyártásának olyan sorrendje, hogy a gyártás össz időtartama a legkisebb legyen.

A módszert egy példán szeretnénk bemutatni. Legyen adott három gép, mégpedig az I., a II. és a III. gép, s azoknak rendelkezésre álló kapacitása, valamint mindegyiken az A, B, C, D, E és F termékeknek a gyártásához szükséges idő.

termékek	gépek		
	I	II	III
A	3	4	5
B	3	3	4
C	1	2	5
D	2	5	2
E	1	2	5
F	3	3	4
rendelkez. álló idő	5	6	7

Ha a termékek adott sorrendben kerülnek feldolgozásra, akkor az egyes gépeket addig terhelik, míg teljesen ki nem használják a kapacitásukat, s utána áttérnek másik gépre, s ott is ugyanezt teszik. A rendelkezésre álló gépidőt természetesen nem lehet túllépni. A gépek össz terhelési ideje javítható, ha összehasonlítjuk az egyes termékeknek az egyes gépeken történő feldolgozási idejét. Ez az összehasonlítás arányok kiszámítása segítségével történik. Mindegyik gyártási időt összehasonlítjuk a legjobb gép terhelési idejével. A következő formula szerint:

$$\text{index} = \frac{\text{bármely feldolgozási idő}}{\text{legjobb feldolgozási idő}}$$

Új táblázatot nyerünk, ahol az I. gépen szükséges gyártási időket vettük alapul, mert az a legjobb gyártási idejű gép.

Az I. gép a legalkalmasabb mindegyik termék termelésére, éppen azért, mert azon vannak a legkisebb termelési idők. A módszer alkalmazásakor a többi gépen, esetünkben a II. és a III. gépen, megkeressük a legkisebb indexeket, és az indexekhez tartozó termékekkel nem terheljük az I. gépet, hanem csakis a II. vagy a III. gépet. Egyforma index esetén a rosszabbik gépet terheljük. Ellenőrizzük a legjobb gépen megmaradt termékek össz gyártási idejét, ami kisebb, vagy egyenlő kell hogy

termékek	gépek		
	I	II	III
A	3	4 (1.33)	⑤ (1.66)
B	3	③ (1.00)	4 (1.33)
C	①	2 (2.00)	5 (5.00)
D	2	5 (2.50)	② (1.00)
E	①	2 (2.00)	5 (5.00)
F	3	③ (1.00)	4 (1.33)
rendelk. álló idö	5	6	7
felhaszn. idö	2	6	7

legyen az össz rendelkezésre álló időnél. Ha ez nem teljesül, akkor az I. gépről továbbra is átvisszük az egyes termékek termelését arra a gépre, melyen kisebb a megfelelő index. A termelési összidő itt 15 óra, a rendelkezésre álló összidő pedig 18 óra. A módszer azonban csak szuboptimális megoldást ad. Ugyanis a kapott megoldás, bizonyos megfontolás után még mindig javítható. Esetünkben, ha az A terméket nem a III., hanem az I. gépen, az F terméket pedig nem a II., hanem a III. gépen gyártjuk, akkor még jobb össz időt érhetünk el. Ezt mutatja a következő táblázat:

termékek	gépek		
	I	II	III
A	③	4	5
B	3	③	4
C	①	2	5
D	2	5	②
E	①	2	5
F	3	3	④
rendelkez. álló idö	5	6	7
felhaszn. idő	5	3	6

Az össz felhasználási idő most 14 óra.

Azzal, hogy figyelembe vesszük, hogy a módszer csak szuboptimális megoldást ad, el kell ismernünk korlátolt lehetőségét. Azonban azáltal, hogy több gépen, több munkának (terméknek) sorbarendezését meg lehet oldani, leszögezhetjük, hogy a módszer e szempontból igen előnyös

tulajdonsággal rendelkezik. A módszer lépései az elmondottak alapján a következők:

1. lépés: Az egyes gépeken megadott gyártási idők alapján, a legjobb gép gyártási idejét véve alapul kiszámoljuk a gyártási idők indexeit, úgy, hogy az egyes feldolgozási időket elosztjuk a legjobb gép feldolgozási idejével.

2. lépés: A legjobb gép kivételével a következő gépeken megkeressük a legkisebb indexeket.

3. lépés: A kijelölt termékeket a legjobb gépről a többi gépre visszük át, oda, ahol legkisebbek az indexek. Egyforma index esetén a terméket a rosszabb gépre visszük át.

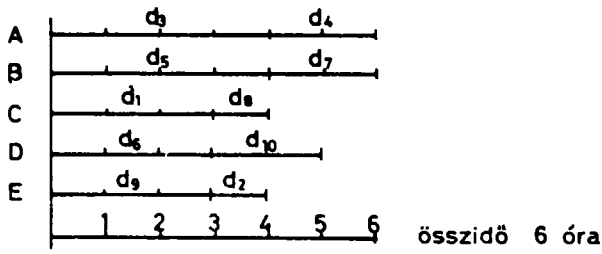
4. lépés: A legjobb gép össz rendelkezésre álló ideje nem léphető át, ezért a 2. és 3. lépést addig ismételjük, amíg annak felhasználási ideje kisebb vagy egyenlő az össz idővel. Természetesen a 2. és 3. lépés ismétlésekor a már elrendezett termékek idejét nem vesszük figyelembe.

6. *Graham algoritmus*[6]

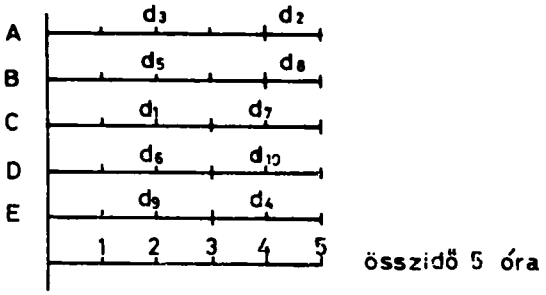
A Graham algoritmus olyan sorbarendezési problémák megoldására alkalmas, melyeknél m gépen n műveletet (munkadarabot) kell sorbarendezni úgy, hogy az össz munkadarab feldolgozása a legrövidebb idő alatt készüljön el. Legyen adott A, B, C, D és E gép, melyeknek különböző átfutási idejű műveleteket kell elvégezni. Mindegy, hogy melyik gépen végezzük el az egyes műveleteket. Az egyes műveleteket jelöljük d_i-vel (i = 1, 2, ... n), s értékük legyen:

$d_1=3$, $d_2=1$, $d_3=4$, $d_4=2$, $d_5=4$, $d_6=3$, $d_7=2$, $d_8=1$, $d_9=3$, $d_{10}=2$.

A feladat megoldásakor ki kell választani egy tetszőleges k értéket ($0 \leqq k \leqq n$) s az adott átfutási időkből az első k leghosszabbikát. Legyen a k adott esetben 6. Így most az első 6 leghosszabb átfutási idő: $d_3=4$, $d_5=4$, $d_1=3$, $d_6=3$, $d_9=3$, $d_4=2$. A kiválasztott k munkadarabot az n gépen sorbarendezzük, adott esetben az első 6 kiválasztott átfutási időt az A, B, C, D és F gépeken. A megmaradt n-k munkadarabot, a d_2, d_7, d_8, és 10_{10} időket tetszőlegesen csoportosítjuk a gépeken. Arra azonban vigyázni kell, hogy ne legyen gép, amely nem dolgozik, és ne legyen olyan munkadarab, amely ne került volna feldolgozásra. A feladat megoldását a következő ábra mutatja:



A kapott össz átfutási idő szuboptimális eredmény. Ugyanis a módszer szuboptimális eredményt ad, amely bizonyos megfontolások alapján javítható, de optimumnak is elfogadható. A javításkor az előzőtől jobb megoldást kaphatunk:



A módszer előnyét messzemenően biztosítja azon tulajdonsága, hogy lehetőség van nagyszámú gépen nagyszámú munkadarab sorbarendezésére.

A módszer lépésekben összefoglalva a következő:

1. lépés: Tetszőleges k-ra ($0 \leqq k \leqq n$) kiválasztjuk a $\{d_j\}$ ($i = 1, 2, \ldots, n$) átfutási idők halmazából az első k leghosszabb átfutási időket.

2. lépés: Az m gépen, a k munkadarabhoz kiválasztjuk az optimális csoportosítást, tetszőleges módszert alkalmazva.

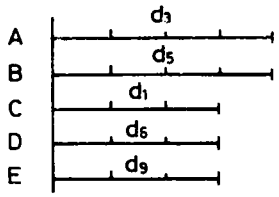
3. lépés: A fennmaradó n-k munkadarabot tetszőlegesen csoportosítjuk a gépen, egyedül csak arra vigyázva, hogy ne legyen gép, amely nem dolgozik, s ugyanakkor ne legyen olyan munkadarab, amely még nem került gépre.

4. lépés: A kapott megoldást, ha lehet, javítjuk vagy optimálisnak fogadjuk el.

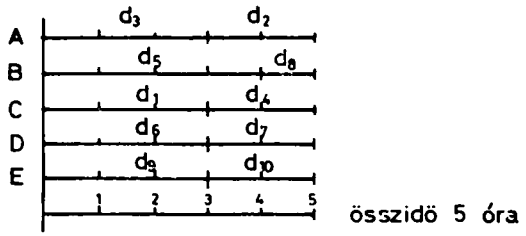
7. Egy új sorbarendezési algoritmus[7]

A módszer a Graham algoritmusra támaszkodik, azzal a különbséggel, hogy itt az egyes lépések szigorúbb követelményeknek tesznek eleget, s ezáltal végeredményként már nem javítható optimális megoldást kapunk. A módszer m gépén n művelet (munkadarab) sorbarendezésére alkalmas. A módszer különösen alkalmas olyan problémák megoldására, melyeknél az n az m-nél néhányszor nagyobb. Legyen szintén adott A, B, C, D és E gép, s 10 termék átfutási ideje: $d_1 = 3$, $d_2 = 1$, $d_3 = 4$, $d_4 = 2$, $d_5 = 4$, $d_6 = 3$, $d_7 = 2$, $d_8 = 1$, $d_9 = 3$, $d_{10} = 2$. A feldolgozás sorrendje nem lényeges, az össz átfutási időt kell minimalizálni. A megoldási menet első lépéseként ki kell választani annyi munkadarabot, ahány gép van, esetünkben $m = 5$, de a legnagyobb átfutási időkkel. A kiválasztott munkadarabokat szétosztjuk, úgy, hogy minden gépre jusson egy. A megmaradt átfutási időkből szintén kiválasztjuk a következő m legnagyob-

bikat. A gépekre történő szétosztás most a következő elv alapján történik: a legnagyobb átfutási idejű terméket arra a gépre irányítjuk, amelyen az előző termék (vagy termékek) befejezési ideje a legkisebb. Ezt addig végezzük, míg a második m munkadarabot is elrendeztük. A megmaradt munkadarabok átfutási idejéből most szintén kiválasztjuk a következő m legnagyobbat, s az előző elv alapján terheljük a gépeket. Esetünkben az első m=5 legnagyobb átfutási idők a következők: $d_3=4$, $d_5=4$, $d_1=3$, $d_6=3$, $d_9=3$.



A következő 5 átfutási idő: $d_2=1$, $d_4=2$, $d_7=2$, $d_8=1$, $d_{10}=2$. Ezen átfutási időket a következőképpen rendezzük el:



A kapott befejezési idő nem javítható. Az optimális befejezési idő egyébként megegyezik a Graham módszer alkalmazásával kapott optimális idővel.

Az algoritmus lépései a következők:

1. lépés: A $\{d_i\}$ $(i=1, 2, \ldots, n)$ átfutási idő halmazából kiválasztjuk az első m leghosszabb átfutási időket.

2. lépés: Az m gépen csoportosítjuk a kiválasztott m munkadarab átfutási idejét.

3. lépés: A fennmaradó n-m munkadarab közül kiválasztjuk a következő m legnagyobb átfutási időket.

4. lépés: Ezen kiválasztott időket úgy csoportosítjuk az n gépen, hogy a legnagyobb átfutási időt arra a gépre irányítjuk, amelyen az eddigi termékek össz ideje a legkisebb. A sorba rakott időt, valamint a gépet töröljük, s újból alkalmazzuk a csoportra a 4. lépés elvét.

5. lépés: A megmaradt n-2m, n-3m, ... munkadarab közül kiválasztjuk a következő m munkadarabot, majd áttérünk a 4. lépésre. Ezeket a lépéseket addig ismételjük, míg el nem fogynak a munkadarabok.

6. lépés: Az ábra alapján kiszámoljuk az össz átfutási időt.

8. A sorbarendezési módszerek összehasonlítása és összehasonlítási kritériumai

A következőkben megadjuk a sorbarendezési módszerek összehasonlítására alkalmazott kritériumokat, s feltüntetjük a kritériumokon belüli intenzitásokat. Természetesen ezzel nem merítettük ki az összehasonlítási kritériumokat, mert ahhoz részletesebb kutatásra lett volna szükség.

1. Az adatgyűjtéskor szükséges befektetés szintje (idő, munka és egyéb befektetés)
   — csekély
   — közepes
   — sok
2. Az adatrendezés és modellezés igénye (előzetes tudás szempontjából)
   — előzetes tudást nem igényel
   — előzetes tudást igényel lineáris algebrából vagy gráfelméletből
   — részletes tudást igényel lineáris algebrából és gráfelméletből
3. A módszer alkalmazásának feltétele és követelménye
   — nem igényel tudást lineáris algebrából és gráfelméletből
   — közepes szintű tudást igényel lineáris algebrából és gráfelméletből
   — magas szintű tudást igényel lineáris algebrából és gráfelméletből
4. Alkalmazási lehetőség általánosság szempontjából
   — szűk (n termék 1 gépen történő sorbarendezése)
   — közepes (n termék 2 gépen történő sorbarendezése)
   — igen széles (n termék vagy művelet m gépen való sorbarendezése)
5. Alkalmazási lehetőség alkalmazási terület szempontjából (közgazdaság, gyártási sorok, munkaszervezés, szociológia, biológia, politika stb.)
   — szűk
   — közepes
   — igen széles
6. A feldolgozás jellege (gépi, kézi vagy mindkét feldolgozás számításba jöhet)
   — csak kézi
   — kézi és gépi
   — csak gépi
7. A számítás terjedelme az optimumhoz jutásig
   — kevés számítást igényel
   — közepesen sok számítást igényel
   — sok számítást igényel
8. Az eredmény pontossága
   — a megoldás csak közelítő, a következő lépésben javítható
   — az eredmény közelítő megoldás
   — megkapjuk a megbízható optimumot (pontosan meghatározott hibakorláttal)
9. Az eredmény struktúrája
   — kizárólag csak a sorbarendezés eredményét kapjuk meg

A SORBARENDEZÉSI MÓDSZEREK ÖSSZEHASONLÍTÓ TÁBLÁZATA

	Foulkes algoritmus	Oszlop-vektoros eljárás	Grafikus eljárás	Johnson algoritmus	Index módszer	Graham algoritmus	Új algo-ritmus
1. A befektetés szintje adatgyűjtéskor	közepes	közepes	közepes	közepes	közepes	csekély	csekély
2. Az adatrendezés és modellezés igénye	előzetes tudást igényel gráfelmél.	előzetes tudást igényel gráfelmél.	előzetes tudást igényel gráfelmél.	előzetes tudást nem igényel	előzetes tudást nem igényel	előzetes tudást nem igényel	előzetes tudást nem igényel
3. A módszer alkalmazásának feltétele és követelménye	közepes tudást igényel lin. alg.	közepes tudást igényel lin. alg.	közepes tudást igényel gráfelmél.	nem igényel előzetes tudást	nem igényel előzetes tudást	nem igényel előzetes tudást	nem igényel előzetes tudást
4. Alkalmazási lehetőség általánosság szempontjából	szűk (1 gép n term.)	szűk (1 gép n term.)	szűk (1 gép n term.)	közepes (2 gép n term.)	igen széles (m gép n term.)	igen széles (m gép n term.)	igen széles (m gép n term.)
5. Alk. lehet. alkalmazási terület szempontjából	közepes	közepes	közepes	közepes	igen széles	igen széles	igen széles
6. A feldolgozás jellege	gépi és kézi	gépi és kézi	csak kézi	gépi és kézi	gépi és kézi	gépi és kézi	gépi és kézi
7. A számítás terjedelme	sok számítást igényel	sok számítást igényel	közepesen sok számítást igényel	közepesen sok számítást igényel	közepesen sok számítást igényel	közepesen sok számítást igényel	közepesen sok számítást igényel

8. Az eredmény pontossága	közelítő megoldás	közelítő megoldás	közelítő megoldás	közelítő megoldás	közelítő megoldás javítható	közelítő megoldás javítható	közelítő megoldás
9. Az eredmény struktúrája	a sorrend mellett más inf. is levez.	csak a sorbarendezés eredménye	csak a sorbarendezés eredménye	a sorrend mellett más inf. is levez.	a sorrend mellett más inf. is levez.	a sorrend mellett más inf. is levez.	a sorrend mellett más inf. is levez.
10. A megoldás menetének tulajdonsága	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre	nincs szükség heurisz. döntésre

— az optimális sorrend mellett más információk is levezethetők
— az optimális sorbarendezés mellett még más eredményt is kapunk

10. A megoldás meneténak tulajdonsága
— a megoldás menete közben heurisztikus döntésre van szükség
— a megoldás folyamán nincs szükség heurisztikus döntésre.

Zárószó

Munkánkban ismertettük a gyakorlatban jelentkező sorbarendezési problémákra viszonylag nehézség nélkül alkalmazható megoldási módszereket, illetve algoritmusokat. A módszereket úgy ismertettük, hogy a műszaki vagy gazdasági szakemberek azonnal alkalmazni is tudják őket. Ezért mellőztük a matematikai bizonyításokat és levezetéseket, ezeket megtaláljuk az irodalmi utalásokban. Természetesen nem merítettük ki az összes módszerek leírását. A szakirodalomban azonban erre a témára ritkán találunk megfelelő anyagot. Munkánkban minden módszernél megadtuk a lépéseket, amelyek folytán az alkalmazás bizonyos szempontból egyszerűbbé válik. A módszerek és algoritmusok bemutatása mellett a munka tartalmazza még a feldolgozott sorbarendezési módszerek összehasonlítását. Tekintettel arra, hogy a módszerek között alkalmazhatóság, számítási igény, előzetes tudás vagy egyszerűség szempontjából igen nagy különbség van, így az összehasonlító táblázat megszerkesztését igen fontosnak tartjuk. Az összehasonlító táblázat alapját azok a kritériumok képezik, amelyek szerint a módszerek összemérhetők, s amelyeket a kritériumok elsősorban minőségi szempontból jellemeznek. Természetesen nem állítjuk, hogy a viszonylagos összehasonlítási kritériumoknak száma és tartalma nem változtatható. Sőt továbbfejlesztésük és bővítésük fontos és érdekes feladat.

Jegyzetek

[1] A Foulkes algoritmusról bővebben lásd: A. Kaufmann: Az operációkutatás módszerei és modelljei c. könyvének 20., valamint 220. oldalán. A könyvet a Közgazdasági és Jogi Könyvkiadó adta ki Budapesten, 1968-ban.

[2] Kaufmann, Deshazeille: A kritikus út módszerének matematikai alapjai. Műszaki Könyvkiadó, Budapest, 1972., 20. oldal.

[3] Kaufmann, Deshazeille: A kritikus út módszerének matematikai alapjai. Műszaki Könyvkiadó, Budapest, 1972., 26. l.

[4] Forgó Ferenc: Nemkonvex és diszkrét programozás. Közgazdasági és Jogi Könyvkiadó, Budapest, 1978., 387. l.

[5] Robert W. Metzger: Elemi matematikai programozás. Gondolat Könyvkiadó, Budapest, 1971., 160. l.

[6] Lásd: Forgó Ferenc Nemkonvex és diszkrét programozás c. könyvét. Közgazdasági és Jogi Könyvkiadó, Budapest, 1978., 383. 1.

[7] A módszerrel eddig még nem találkoztunk. Értelmi szerzője Mészáros Katalin, az algoritmus részletesebb leírását Molnár S. Verona fogalmazta meg.

Rezime

Nekoliko metoda vremenskog raspoređivanja

U svakodnevnoj ekonomskoj praksi često se javljaju problemi vremenskog raspoređivanja aktivnosti. Za ove probleme je karakteristično da se aktivnosti mogu realizovati u različitom redosledu, ali koji treba da zadovolje unapred postavljene zahteve. Postoje različiti metodi za rešavanje problema ovakvih vrsta, ali nije svaki metod podjednako efikasan za svaki problem vremenskog raspoređivanja. U ovom radu je dat kratak prikaz, kao i kraći postupak rešavanja nekoliko algoritma. Za njihovu primenu nije potrebno detaljnije matematičko-stručno znanje. Zatim u jednoj tabeli obuhvaćeno je relativno vrednovanje metoda na bazi upoređivanja istih, kao i njihova klasifikacija.

Summary

Some Methods of Time Distribution

In economic practice, often appears the problem of time distribution. The major characteristic of this problem is that the activities can be realised in different order. There are few methods of putting the activities in order, each of them is equeally statisfactory. This work gives a brief outline and short proccess description of algorythm solvation. Their application doesn't demand special mathematical and professional knowledge. There is a list of relation valuation of methods, and their classification.