Optimalizálási Rendszerek és Matematikai Modellezés

példákon keresztül

G.-Tóth Boglárka

Tartalomjegyzék

Előszó

Napjainkban az optimalizálás – ami lehet gyártástervezés, költség minimalizálás, profit maximalizálás, stb. – az élet bármely területén megjelenő feladat. Ezek a problémák általában nagyon sok tényezőtől függnek, és a valóságnak teljesen megfelelő felírása nem lehetséges, egyrészt a szükséges adatok pontosságának hiánya miatt, másrészt az így adódó feladat bonyolultsága miatt. Természetesen ezek a kijelentések nem jelentik azt, hogy így aztán ezekkel a feladatokkal nem kell foglalkozni. Sőt. Az igazi kihívás az adott probléma olyan matematikai felírása – ezt modellezésnek nevezzük – ami a legfontosabb tényezőket figyelembe veszi, és a bonyolultsága még éppen akkora, hogy megoldható legyen. Ezzel a két legmeghatározóbb feladatot definiáltuk is: egy feladat jó matematikai felírása, és a matematikai modell megoldása.

Szerencsére a problémák nagy része felírható lineáris programozási feladatként (LP), azaz a változók lineáris kombinációjából álló célfüggvényt optimalizáljuk egy síkokkal határolt konvex halmazon, vagyis a változókban lineáris feltételrendszer mellett. Ezekben az esetekben azért van szerencsénk, mert az LP feladatok polinomiális időben megoldhatók, vagyis a futási idő a változók számától polinomiálisan függ. Ráadásul LP feladatok megoldására számos jól kifejlesztett megoldó (solver) létezik, – ezek közül a legelterjedtebbek használatát be is fogjuk mutatni, – és a mai korszerű számítógépek használatával ez százezres, de akár milliós változószámú és feltételszámú feladatok megoldását is lehetővé teszi.

Azok a problémák, amelyek nem írhatóak fel lineáris alakban, szükségképpen csak nemlineáris függvényekkel fogalmazhatók meg. A nemlinearitás még nem feltétlen jelent rosszat. Ha a probléma csak néhány változójában nemlineáris, vagy a nemlinearitás egyszerű, pl. szorzat, akkor sokszor átírható lineáris alakba, persze több változóval és feltétellel. Másrészt, ha a függvényünk kvadratikus, vagy konvex akkor jó esélyünk van megtalálni az optimumot speciális nemlineáris módszerekkel rövid időn belül. Multimodális, azaz több lokális optimummal rendelkező feladatok esetén viszont NP-nehéz a globális optimalizálási feladat, és nagy változószám esetén csak közelítő megoldás megtalálására van esélyünk.

A jegyzet célja, hogy az olvasót bevezesse a matematikai modellezés nehézségeibe és szépségeibe, miközben megismerteti a leggyakrabban használt modellező nyelvekkel és megoldó szoftverekkel. Az első három fejezetben a modellezésen lesz a

hangsúly, megmutatjuk pár alapvető feladat modelljét, illetve bevezetjük az olvasót néhány felírási és átírási trükk használatába is. A következő három fejezetben további példákon keresztül megtanuljuk pár széles körben elterjedt modellezési nyelv rejtelmeit, miközben gyakoroljuk a modellezést. A 7. fejezetben röviden bemutatjuk a használható megoldókat, lehetséges beállításaikat, módszereiket, végül feladatokkal és megoldásaikkal zárjuk mondandónkat.

A jegyzet központi mondandójának követéséhez nem szükséges a lineáris algebra alapjain kívül nagyobb felkészültség, de a mélyebb megértést sokban segíti az Operációkutatás, pontosabban a Lineáris programozás alapvető ismerete. Így ez az összefoglaló hasznos lehet nem csak matematikus, és informatikus hallgatók, kutatók számára, hanem egyaránt bármely, az alkalmazási oldalról érkező érdeklődő olvasónak. Ahol lehetséges volt, példák segítségével próbáltuk segíteni a megértést, illetve sok helyen feladatokkal látjuk el az olvasót, hogy próbára tegye a megszerzett tudását.

Kívánom, hogy ez az összefoglaló az optimalizálási programrendszerek iránt érdeklődők segítségére legyen és olvasóim haszonnal forgassák.

G.-Tóth Boglárka

1. fejezet

Bevezető

Ahogy az előszóban is említettük, az első feladat a matematikai modell felírása, aminek minél jobban illeszkednie kell a valós problémához, mindamellett a lehető legegyszerűbb feladatot szeretnénk kapni a megoldhatóság miatt. Nézzünk egy példát az érthetőség kedvéért.

1.1. Példa. A Kefe Zrt. üzemében 4 féle kefefejet készítenek: K-1, K-2, K-3, és K-4. A kefefejek műanyag sörtékből készülnek, amelyek előállításához különböző minőségű alapanyagokat szoktak beszerezni. Jelenleg az I. osztályúból 22 600 kg, a II. osztályúból 25 400 kg, míg a III. osztályúból 2 600 kg áll rendelkezésre.

A különböző kefefejek természetesen különböző összetételűek. A következő táblázat tartalmazza, hogy a különböző kefefejfajták esetén 100 darab elkészítéséhez mennyi és milyen minőségű műanyag szükséges.

1.1. táblázat. A különböző kefefajták alapanyagszükséglete.

	Kefefajták			
Sörték	K-1	K-2	K-3	K-4
I. osztályú	9	2	3	0
II. osztályú	4	7	3	4
III. osztályú	1	1	0	2

Az egyes kefefajták darabjának nyeresége a Kefe Zrt.-nek rendre 30, 22, 13 és 10 Ft. Hány darabot készítsen az üzem a különféle kefefejekből, ha a maximális nyereség elérését tűzték ki célul?

Megoldás: A matematikai modell felírása viszonylag egyszerű, hiszen a feladatban csak olyan információt adtunk meg, amit fel is kell használnunk. A modell felírásának első és egyik legfontosabb feladata a (döntési) változók meghatározása. Legtöbbször a célkitűzés segítségével lehet ezt a legkönnyebben meghatározni. Vegyük például az utolsó mondatot:

> *Hány darabot készítsen az üzem a különféle kefefejekből, ha a maximális nyereség elérését tűzték ki célul?*

Vagyis arról kell döntenünk, hogy hány darab kell a kefefejekből, így ezeket célszerű változóknak választani. Legyen tehát x_1 a gyártandó K-1 kefefejek száma, míg rendre x_2 a K-2, x_3 a K-3, x_4 a K-4 kefefejek száma. A fenti mondatból az optimalizálandó célfüggvényt is könnyű felírni. Ha x_1, x_2, x_3, x_4 az egyes kefefejek gyártandó száma, akkor ehhez a termelési tervhez tartozó nyereség $30x_1 + 22x_2 + 13x_3 + 10x_4$. Tehát a célfüggvény, amit maximalizálunk a

$$30x_1 + 22x_2 + 13x_3 + 10x_4.$$

Könnyen beláthatjuk, hogy feltételrendszer nélkül ennek a függvénynek nincs véges maximuma, másképp fogalmazva nem korlátos a feladat.

A feltételeket az egyes alapanyagok korlátozott mennyisége adja. Egy termelési terv csak akkor kivitelezhető, ha a felhasználandó alapanyagok mennyisége nem lépi túl a rendelkezésre álló mennyiséget, azaz,

$$\begin{aligned} 9x_1 + 2x_2 + 3x_3 &\leq 22600 \\ 4x_1 + 6x_2 + 3x_3 + 4x_4 &\leq 25390 \\ x_1 + x_2 + 2x_4 &\leq 1800 \end{aligned}$$

A figyelmes olvasónak feltűnhet, hogy az 1.1. táblázatban az alapanyagok 100 db kefe elkészítéséhez adottak. Vagyis a fenti feltételek csak akkor igazak, ha a változóink az egyes kefefajták mennyiségét 100 darabjával adják meg, és így a célfüggvényünk a következőre változik: $\max 3000x_1 + 2200x_2 + 1300x_3 + 1000x_4$. Persze ha nem akarjuk, hogy az együtthatóink túl nagyok legyenek, átírhatjuk ezer forintra is a célfüggvényt:

$$\max 3x_1 + 2,2x_2 + 1,3x_3 + x_4. \tag{1.1}$$

Visszatérve a feltételekre, természetesen nem feledkezhetünk meg a változók jelentéséből adódó nemnegativitási feltételekről sem, vagyis

$$x_i \geq 0, i = 1, \ldots, 4.$$

Összefoglalva a probléma matematikai modellje felírható egy lineáris programozási feladatként, ami a következő.

$$\begin{array}{rrrrl} 9x_1 + & 2x_2 + & 3x_3 & \leq 22600 & \\ 4x_1 + & 6x_2 + & 3x_3 + 4x_4 & \leq 25390 & \\ x_1 + & x_2 + & 2x_4 & \leq 1800 & \\ & & x_i & \geq 0, \ \forall\, i & \\ \hline \max 3x_1 + 2, & 2x_2 + 1, & 3x_3 + & x_4 & \end{array} \tag{1.2}$$

Ezt a feladatot megoldhatjuk, például, a szimplex módszer segítségével. A megoldás az $x = (1020, 1580, 3420, 0)$ termelési terv 10 982 eFt haszonnal, vagyis 102 000 K-1, 158 000 K-2, 342 000 K-3 és 0 K-4 termelése mellett szerezzük meg a maximális profitot.

A fenti példa egy egyszerű gyártástervezési feladat volt, ahol a feladatban megadott információ pontosan elegendő volt a matematikai modell felírásához, és látszólag a megoldáshoz nincs is szükség másra. Gondolkodjunk el viszont a következő kérdéseken:

- Mekkora a kereslet az egyes kefefejekre?
- Mennyi munkaerő illetve géperő kell az egyes kefefajták előállításához?
- A rendelkezésre álló alapanyagok beszerzési árai figyelembe lettek véve a számolt nyereségekben?
- Nem lenne célszerű lehetőséget adni további alapanyagok beszerzésére?

Természetesen azt feltételezzük, hogy a fenti feladat úgy lett megfogalmazva, hogy minden fontos információt tartalmazzon, ezek a kérdések csak arra akarnak rámutatni, hogy egy valós probléma esetén sok olyan kérdés merül fel, ami nehezíti a modellező munkáját egy egyszerű modell felírásában.

1.2. Példa. Most tételezzük fel, hogy az utolsó kérdésünkre igen a válasz. Legyen 300, 250 és 200 forint rendre az I., II. és III. osztályú alapanyagok kilónkénti beszerzési ára. Hogyan módosul az (1.2) modellünk?

Megoldás: Az (1.2) modellben a feltételek csak a raktárban lévő alapanyagokat vették számításba. Ha hozzá akarjuk venni a lehetőséget, hogy plusz alapanyagokat vásároljunk, akkor módosítanunk kell a feltételek jobboldalát. Vegyük észre, hogy minden alapanyagra egy új változót kell definiálnunk, ami megadja, hány kilót vásárolunk az

adott alapanyagból. Legyenek ezek $y_1, \ldots, y_3$ a három alapanyagra. Így a feltételeink egyszerűen az

$$\begin{aligned} 9x_1 + 2x_2 + 3x_3 &\leq 22600 + y_1 \\ 4x_1 + 6x_2 + 3x_3 + 4x_4 &\leq 25390 + y_2 \\ x_1 + x_2 + 2x_4 &\leq 1800 + y_3 \end{aligned}$$

feltételekre módosulnak. Viszont így felmerül a kérdés: Hol vegyük figyelembe az alapanyagok beszerzéséből keletkező költségeket? Az első gondolatunk lehet, hogy az előző (1.1) célfüggvényből vonjuk ki a $300y_1 + 250y_2 + 200y_3$ költséget. Ez sajnos jelenleg két ok miatt is rossz. Az (1.1) célfüggvényben a nyereség ezer forintokban adott –ezt azért könnyű orvosolni–, és itt az egyes kefék nyeresége van megadva, ami azt jelenti, hogy ha ezek jól számolt értékek, akkor tartalmazzák az alapanyagok árát, a megmunkálás költségét, stb. Vagyis ide nem tudjuk felvenni, és esetleg meglepő módon, de nem is kell felvenni az újonnan vásárolt alapanyagok költségeit. Nézzük meg akkor, hogy így milyen modellhez jutottunk.

$$\begin{aligned} 9x_1 + 2x_2 + 3x_3 &\leq 22600 + y_1 \\ 4x_1 + 6x_2 + 3x_3 + 4x_4 &\leq 25390 + y_2 \\ x_1 + x_2 + 2x_4 &\leq 1800 + y_3 \\ x_i, y_j &\geq 0, \ \forall\, i, j \\ \hline \max 3x_1 + 2, 2x_2 + 1, 3x_3 + x_4 & \end{aligned} \tag{1.3}$$

Mivel y_i nincs korlátozva, a feltételek jobboldalai lényegében bármeddig nőhetnek, így a felírt modellünk nem korlátos. Gondolhatjuk, hogy valamit elrontottunk, de az adott információk mellett nem tudunk jobbat felírni.

Ennek a példának az a fontos mondanivalója, hogy sokszor nem csak a megadott adatokra kell figyelni, hanem arra is, hogy azok összefüggései is adottak legyenek. Ebben a példában, hogy „értelmes" modellt kapjuk, további információra van szükség. Például mik a gyártási korlátok, azaz a munkaerő, gépkapacitás, raktár, vagy egyéb korlátozás, illetve mik a piac nyújtotta megkötések a keresletre vonatkozóan. Amíg ezekre vonatkozó adatokat nem kapunk, a modellt csak az eredeti verzióban írhatjuk fel, ahol csak a meglévő alapanyagokból dolgoztunk.

2. fejezet

Általános modellek

Ebben a fejezetben bemutatjuk azokat a standard modelleket, amik alapvetőek a modellezési feladatok megoldásához. Az első három alfejezet feladatai ráadásul olyan modellekre vezetnek, amelyek speciális módszerekkel megoldhatók.

2.1. Gyártástervezés

A gyártástervezési feladatok adják a legegyszerűbb és ezáltal legérthetőbb modelleket, ezért is szokás ezekkel a problémákkal kezdeni. Tulajdonképpen az első fejezetben is már egy ilyen feladatot írtunk fel, adott termékek előállítását terveztük meg a megadott alapanyagokból. Itt még sokszor előjönnek olyan feltételek, amik nem materiális alapanyagokra vonatkoznak, hanem például munkaerő kapacitásra, vagy egy használandó gép kapacitására, de akár a késztermékek tárolására is. Ezek mind-mind ugyanolyan típusú feltételre vezetnek, azaz az előállítandó termékek egy lineáris összetétele kisebb vagy egyenlő mint valamely erőforráskorlát. De lássunk erre egy példát.

2.1. Példa. Egy asztalosműhelyben székeket, asztalokat, padokat és polcokat gyártanak. A műhelyben egy munkás van a famegmunkálásra, egy végzi a bútorok összerakását, és egy pedig a felületkezelését. Az egyes bútorok munkaigényét, és a szükséges anyagokat a 2.1. táblázat adja meg, ahol szintén láthatjuk a rendelkezésre álló mennyiségeket egy egy hetes periódushoz.

A kérdés magától értetődő: A rendelkezésre álló erőforrások mellett miből mennyit gyártsunk, hogy maximalizáljuk a bevételt?

Megoldás: A matematikai modell felírása megint a változók meghatározásával kezdődik. A kérdésből ez ismét könnyen leolvasható: „miből mennyit gyártsunk”, vagyis az egyes termékek mennyiségei lesznek a változóink. Jelöljük ezeket $x_1, \ldots, x_4$-el, a felsorolásunknak megfelelően a székek, asztalok, padok és polcok gyártandó mennyiségei.

2.1. táblázat. Adatok a 2.1. példához

Erőforrások	Termékek				Felhasználható
	szék	asztal	pad	polc	
famegmunkálás (óra)	1,5	1	1,5	0,3	40
bútorösszerakás (óra)	1	0,5	1,5	0,2	40
felületkezelés (óra)	1	1	1,5	0,3	40
faanyag (m^3)	0,05	0,1	0,1	0,05	2
csavar (db)	0	4	10	0	72
ragasztó (liter)	0,5	0,2	0,3	0,3	10
festék (liter)	0,5	0,5	0,7	0,4	15
Ár (ezer forint)	5	10	20	3,5	

A megadott táblázat nagyon egyszerűvé teszi a modell megadását, hiszen az utolsó kivételével minden sor egy feltételnek felel meg, míg az utolsó adja meg a maximalizálandó célfüggvényt, azaz a modell

$$\begin{aligned}
1,5x_1 + \quad x_2 + 1,5x_3 + \; 0,3x_4 &\leq 40 \\
x_1 + 0,5x_2 + 1,5x_3 + \; 0,2x_4 &\leq 40 \\
x_1 + \quad x_2 + 1,5x_3 + \; 0,3x_4 &\leq 40 \\
0,05x_1 + 0,1x_2 + 0,1x_3 + 0,05x_4 &\leq \; 2 \\
4x_2 + \; 10x_3 \qquad\quad &\leq 72 \\
0,5x_1 + 0,2x_2 + 0,3x_3 + \; 0,3x_4 &\leq 10 \\
0,5x_1 + 0,5x_2 + 0,7x_3 + \; 0,4x_4 &\leq 15 \\
x_i &\geq 0, \forall\, i \\
\hline
\max 5x_1 + \; 10x_2 + \; 20x_3 + \; 3,5x_4 &
\end{aligned} \tag{2.1}$$

Ha a 2.1. táblázatban a felhasználható mennyiségeket egy b vektorba, míg az gyártási tényezőket (azaz a táblázat középső részét) az A mátrixba tesszük, akkor a fenti feltételrendszer az $Ax \leq b, x \geq 0$ alakra redukálódik. Ez azt is jelenti, hogy akár az egyszerű szimplex módszerrel is megoldható.

Ennek a feladatnak az Excel Solverrel való megoldását tárgyaljuk majd a 4. fejezetben.

2.2. táblázat. Költségek a 2.2. példa gyártástervezési feladatához

Erőforrások	Költség
Famegmunkálás (óra)	700 Ft
Bútorösszerakás (óra)	800 Ft
Felületkezelés (óra)	600 Ft
Faanyag (m^3)	35 000 Ft
Csavar (db)	1 000 Ft
Ragasztó (liter)	500 Ft
Festék (liter)	7 000 Ft

A 2.1. példában minden a legegyszerűbb formában volt adott. Most gondoljuk végig, hogyan változik a modell, ha a rendelkezésre álló erőforrásokhoz költség is tartozik, azaz a célfüggvényünk nem egyszerűen a bevétel, hanem a nyereség maximalizálása.

2.2. Példa. Legyenek adottak a 2.1. példa adatai. Oldjuk meg a feladatot profit maximalizálással, ha a munkások az elvégzett munka után kapnak órabért, és adott a nyersanyagok egységnyi ára is. Ezeket az adatokat a 2.2. táblázatban találjuk.

Megoldás: A költségek alapján az egyes áruk nyereségét könnyedén megadhatjuk, hiszen az árból ki kell vonnunk az egy termékre eső költségeket. Ha c_j-vel jelöljük a j erőforrás költségét, és f_{ij}-vel az i termék j erőforrás felhasználását, akkor az i termék költsége $\sum_j c_j \cdot f_{ij}$. Így az i termék fajlagos nyeresége

$$p_i = a_i - \sum_j c_j \cdot f_{ij}$$

ahol a_i az i termék ára.

Tehát a modellünk csak a célfüggvényben változik, ami $\max \sum_i p_i \cdot x_i$, vagyis pontosan

$$\max 0,2x_1 + 0,35x_2 + 2,71x_3 + 0,77x_4.$$

2.2. Szállítási feladat

2.3. Példa. Három raktár (S1,S2,S3) szolgálja ki négy diszkont (T1,T2,T3,T4) igényeit. A raktárak kapacitása és a diszkontok megrendelése (megfelelő mértékegységekben)

a 2.3. táblázatban vannak megadva, míg az egységnyi szállítás ára a diszkontokra minden raktárból 2.4. táblázatban látható.

2.3. táblázat. A raktárak kapacitása és a diszkontok megrendelése

Raktárak	S1	S2	S3
Kapacitások	135	56	93

Diszkontok	T1	T2	T3	T4
Megrendelések	62	83	39	91

2.4. táblázat. Szállítási költségek

	T1	T2	T3	T4
S1	132	-	97	103
S2	85	91	-	-
S3	106	89	100	98

Hogyan teljesítsük az összes megrendelést minimális költséggel és a raktárak kapacitásának betartásával?

Megoldás: A matematikai modell felírásához adjuk meg a döntési változókat. Ha ez elsőre nem kézenfekvő, akkor nézzük a célfüggvényt, ami a szállítási költség minimalizálása. Vagyis minden c_{ij} szállítási költséghez tartozik egy változó, x_{ij}, ami azt mondja meg, hogy az adott útvonalon mennyi árút szállítunk. Ekkor a célfüggvény

$$\min \sum_{i,j} c_{ij} x_{ij},$$

ahol i a raktárak, j a diszkontok indexe. A feltételeket is a kérdésben találjuk, azaz „Hogyan teljesítsük az összes megrendelést a raktárak kapacitásának betartásával?”

A megrendeléseket akkor teljesítjük, ha minden diszkontra az odaszállított mennyiségek összege megegyezik a megrendelt mennyiséggel, azaz

$$\sum_{i} x_{ij} = d_j \quad \forall j,$$

ahol most d_j jelöli a j. diszkont megrendelését. A raktárak kapacitásának betartása hasonló, az elszállított mennyiségek összege nem lépheti túl a raktár kapacitását, s_i-t, azaz

$$\sum_j x_{ij} \leq s_i \quad \forall i.$$

A modellből már csak a nemnegativitási feltétel hiányzik, ami jelen esetben nagyon fontos, hiszen elhagyásával a modell nemkorlátossá válik. Vagyis a teljes szállítási modell a következő:

$$\begin{aligned} \sum_i x_{ij} &= d_j \quad \forall j \\ \sum_j x_{ij} &\leq s_i \quad \forall i \\ x_{ij} &\geq 0 \quad \forall i,j \\ \hline \min \sum_{i,j} c_{ij} x_{ij} & \end{aligned}$$

Ezt a modellt fogjuk egy összetett feladat esetén kibővíteni a 3.2. alfejezetben, illetve megmutatni, hogy hogyan lehet felírni hasonló formában. A szállítási feladatot a szállítási szimplex, vagy más néven disztribúciós módszerrel, vagy akár a magyar módszer segítségével is megoldhatjuk. Mivel ezek a módszerek kihasználják a feladat speciális szerkezetét, így sokkal hatékonyabban meg tudják oldani, mint a szimplex módszer általános módozatai.

2.3. Hozzárendelési feladat

Ezekben a feladatokban mindig két halmaz elemeit rendeljük egymáshoz. Gondolhatunk táncpartnerek kiválasztására, munkák gépekhez való hozzárendelésére, de akár hallgatók tételhúzására is. Mint eddig, most is egy példán keresztül írjuk fel a matematikai modellt.

2.4. Példa. Négy gépen kell négy különféle műveletet elvégezni. Minden gép alkalmas mindegyik művelet elvégzésére, de a gépek beállításának lassúsága miatt egy gép csak egy feladatot láthat el. Az egyes költségadatokat a 2.5. táblázat mutatja. Hogyan osszák el a munkákat a gépek között, hogy az összes termelési költség minimális legyen?

Megoldás: A fenti probléma alapvetően hasonlít a szállítási feladatra, akár olyan megfogalmazást is adhatnánk, hogy melyik gép „viszi el" melyik műveletet. A fontos különbség, hogy amíg a szállítási feladatnál változó mennyiségeket vittünk két pont

2.5. táblázat. Az egyes műveletek költségei az adott gépeken.

	Művelet			
Gép	A	B	C	D
I. gép	5	7	5	3
II. gép	4	1	3	7
III. gép	6	7	5	3
IV. gép	2	2	1	4

között, itt bináris a döntés, viszi, vagy sem. Tehát ha a szállítási feladathoz analóg módon akarjuk felírni a modellt, akkor az x_{ij} változó mondja meg hogy az i. gép a j. munkát végzi-e (1 ha igen, 0 ha nem). Ha a költségmátrix elemeit itt is c_{ij}-vel jelöljük, akkor a célfüggvényünk ugyanaz lesz:

$$\min \sum_{i,j} c_{ij} \cdot x_{ij}$$

A feltételeink is nagyon hasonlóak, csak a jobboldalak egyszerűsödnek le a kapacitás és megrendelési mennyiségekről egyre, vagyis

$$\sum_i x_{ij} = 1 \quad \forall j,$$
$$\sum_j x_{ij} \leq 1 \quad \forall i.$$

Itt az első feltételt úgy fogalmazhatjuk meg, hogy minden munka pontosan egyszer legyen elvégezve, a másodikat pedig, hogy minden gép legfeljebb egy feladatot láthat el. Itt a fenti feladatra persze egyenlőség is állhat, de néha több gép adott mint feladat, így ezt az általános felírást hagytuk meg.

Összefoglalva ismét a részleteket, a hozzárendelési feladat általános modellje

$$\begin{aligned} \sum_i x_{ij} &= 1 && \forall j \\ \sum_j x_{ij} &\leq 1 && \forall i \\ x_{ij} &\in \{0,1\} && \forall i,j \\ \hline \min \sum_{i,j} c_{ij} x_{ij} & & & \end{aligned}$$

Ezeket a modelleket a magyar módszer segítségével lehet a leghatékonyabban megoldani. A magyar módszerről részletesen Operációkutatás jegyzetben olvashat az érdeklődő olvasó.

3. fejezet

Felírási-átírási trükkök

Sok esetben nem tudjuk a problémánkat valamely standard modell szerint felírni. Ilyenkor próbálkozhatunk azzal, hogy vesszük a probléma egy lehetséges (de nem standard) modelljét, és ezt megpróbáljuk átírni standard alakra, általában új változók bevezetésével. Nézzük a következő egyszerű alsókorlátos példát erre.

3.1. Alsókorlátos feladat

3.1. Példa. Az 1.1. példát egészítsük ki a következővel: Hogyan változik a felírás és a megoldás, ha a választék megtartása érdekében a Kefe Zrt. úgy dönt, hogy a K-1-ből legalább 50 000 db-ot, a K-2-ből legalább 100 000 db-ot, a K-3-ból legalább 300 000 db-ot, a K-4-ből pedig legalább 30 000 db-ot kell az üzemnek gyártania?

Megoldás: Induljunk ki az 1.1. példa (1.2) gyártástervezési felírásából. Tulajdonképpen nincs más dolgunk, csak néhány feltételt hozzávenni a modellhez. Nevezetesen legyen $x_1 \geq 500, x_2 \geq 1000, x_3 \geq 3000, x_4 \geq 300$, hiszen ne felejtsük el, hogy a változóink a termékeket 100 darabjával jelölték.

Fel lehet-e írni ezt a modellt esetleg kevesebb feltétellel? Az első meglátásunk lehet az, hogy az új feltételek mellett nincs szükség a korábbi nemnegativitási feltételekre. Viszont lineáris programozásból tudjuk, hogy ezek a feltételek szükségesek a szimplex módszer alkalmazásához, így elhagyásukkal nem nyerünk semmit. Másrészt a plusz feltételeink mind "$\geq$" feltételek, ami miatt kétfázisú szimplex vagy módosított szimplex módszert kell alkalmazni. Átírható-e a fenti feladat úgy, hogy a feltételrendszere standard, azaz "$Ax \leq b, x \geq 0$" alakú legyen? Legyen ez az igazi feladatunk.

3.2. Példa. Írja fel a

$$9x_1 + \quad 2x_2 + \quad 3x_3 \qquad \leq 22600$$

$$\begin{array}{rrrl} 4x_1 + & 6x_2 + & 3x_3 + 4x_4 & \leq 25390 \\ x_1 + & x_2 + & 2x_4 & \leq 1800 \\ & & x_1 & \geq 500 \\ & & x_2 & \geq 1000 \\ & & x_3 & \geq 3000 \\ & & x_4 & \geq 300 \\ & & x_i & \geq 0,\ \forall\, i \\ \hline \max 3x_1 + 2, 2x_2 + 1, 3x_3 + & & x_4 & \end{array}$$

lineáris programozási feladatot úgy, hogy a feltételrendszere „$Ax \leq b, x \geq 0$” alakú legyen.

Megoldás: A feladat megoldásához új változókat kell bevezetni az x_i-k helyett. Legyenek ezek y_i-k, és függjenek úgy x_i-től, hogy az alsókorlát feltételek nemnegativitási feltételt jelentsenek y_i-re. Vagyis, ha $x_1 \geq 500$, akkor $x_1 - 500 \geq 0$, és így az $y_1 = x_1 - 500$ választással pont az $y_1 \geq 0$ feltételt kaptuk meg. Általánosan, ha az alsókorlátokat k_i-vel jelöljük, akkor az $y_i = x_i - k_i$ egyenlőséggel adhatjuk meg az új változókat. Ekkor az eddigi $Ax \leq b$ feltételünk a következőképpen alakul:

$$Ax \leq b \quad \Leftrightarrow \quad A(y+k) \leq b \quad \Leftrightarrow \quad Ay + Ak \leq b \quad \Leftrightarrow \quad Ay \leq b - Ak$$

ahol a $b - Ak$ tulajdonképpen az erőforrásokból megmaradó mennyiség a minimálisan legyártandó mennyiség után. Vegyük észre, hogy ha a $b - Ak$ vektor valamely eleme negatív, akkor a feladatnak nincs lehetséges megoldása. Ez mutatja, hogy már a minimálisan előállítandó mennyiség sem lehetséges a megadott alapanyagokból, erőforrásokból.

3.2. Összetett szállítási feladat

Most nézzük meg, hogy mi történik, ha egyszerű szállítási feladat helyett egy kibővített feladatot kapunk. Ha az új feladat felírható mégis egyszerű szállítási feladatként, akkor az eredeti feladat is megoldható a disztribúciós vagy magyar módszer segítségével.

3.3. Példa. Vegyük a 2.3. példát, ami egy egyszerű szállítási feladat volt. Hogyan változik a modell és a megoldás ha történhet szállítás raktárak között és diszkontok között is a 3.1. táblázat szerint megadott költségekkel.

Megoldás: Két új modellt fogunk mutatni. Az első intuitívan egyszerűbb, a második viszont egy egyszerű szállítási modell lesz. Az első, mindkét modellre vonatkozó

3.1. táblázat. Szállítási költségek az új utakra.

Honnan → hova	Költség
S1 → S3	55
S1 → S2	20
S2 → S1	35
S3 → S2	25
T2 → T1	15
T3 → T2	12
T2 → T3	12
T1 → T3	20
T3 → T1	25
T1 → T4	10

változtatás, hogy a több szállítási lehetőség, több változót is jelent, azaz minden új útvonalhoz hozzárendelünk egy új változót.

Az első modellhez vegyük alapul az eredeti feladat megoldását.

$$\begin{aligned} \sum_i x_{ij} &= d_j \quad \forall j \\ \sum_j x_{ij} &\leq s_i \quad \forall i \\ x_{ij} &\geq 0 \quad \forall i,j \\ \hline \min \sum_{i,j} c_{ij} x_{ij} & \end{aligned}$$

Az első feltételt megnézve azt láthatjuk, hogy mind a jobb, mind a baloldalt ki kell egészíteni, hiszen a diszkontokba bejövő élekből már több van, illetve a jobboldalon a megrendelt mennyiséghez hozzá kell adni az elszállított mennyiséget is. A pontosság kedvéért jelöljük I-vel a raktárak halmazát, és J-vel a diszkontok halmazát. A fenti modellben nem írtuk ki, de mindenhol $i \in I$ illetve $j \in J$ volt. Most a megrendelés kielégítésének feltétele

$$\sum_{i \in I} x_{ij} + \sum_{i \in J} x_{ij} = d_j + \sum_{k \in J} x_{jk} \quad \forall j \in J.$$

Hasonlóan, a raktárkapacitás feltétele

$$\sum_{j\in J} x_{ij} + \sum_{j\in I} x_{ij} \leq s_i + \sum_{l\in I} x_{li} \quad \forall i \in I.$$

Ezzel készen is volnánk, a teljes modellt így írhatjuk fel:

$$\begin{aligned}
\sum_{i\in I\cup J} x_{ij} - \sum_{k\in J} x_{jk} &= d_j \quad \forall j \in J \\
\sum_{j\in J\cup I} x_{ij} - \sum_{l\in I} x_{li} &\leq s_i \quad \forall i \in I \\
x_{ij}, x_{jk}, x_{li} &\geq 0 \quad \forall i,l \in I, j,k \in J \\
\hline
\min \sum_{i,j} c_{ij} x_{ij}
\end{aligned}$$

Ez teljesen rendben van, de nem alkalmazható rá a disztribúciós vagy magyar módszer, mivel nem illik az egyszerű szállítási feladat felírására. Ez pedig a megoldás hatékonyságát rontja.

Nézzünk akkor egy más megközelítést. Tegyük kiegyensúlyozottá a feladatot, és nevezzük M-nek az összkeresletet. Azokat a kínálati pontokat, azaz raktárokat, amelyekbe lehet szállítani, vegyük fel keresleti pontként is, azaz diszkontként, és a kereslete (megrendelése) legyen M, kínálata viszont legyen M plusz az eredeti kínálata. Ugyanígy a keresleti pontokat (diszkontokat), amelyekből lehet szállítani, vegyük fel kínálati pontként is (raktárként), és a kínálata legyen M, míg a kereslete legyen M plusz az eredeti kereslete. A kapott modell a következőképpen írható fel.

$$\begin{aligned}
\sum_{i\in I\cup J} x_{ij} &= d_j + M \quad \forall j \in J \\
\sum_{k\in J} x_{jk} &= M \quad \forall j \in J \\
\sum_{j\in J\cup I} x_{ij} &= s_i + M \quad \forall i \in I \\
\sum_{l\in I} x_{li} &= M \quad \forall i \in I \\
x_{ij}, x_{jk}, x_{li} &\geq 0 \quad \forall i,l \in I, j,k \in J \\
\hline
\min \sum_{i,j} c_{ij} x_{ij}
\end{aligned}$$

Ebből tulajdonképpen következik az előző modell ha jobban megnézzük.

3.2. táblázat. A kibővített szállítási feladat adattáblája.

Kínálati p.	Keresleti pontok (diszkontok)								Raktár
(raktárak)	S1	S2	S3	T1	T2	T3	T4	Fiktív	kínálat
S1	0	20	55	132	–	97	103	0	135+M
S2	35	0	–	85	91	–	–	0	56+M
S3	–	25	0	106	89	100	98	0	93+M
T1	–	–	–	0	–	20	10	0	M
T2	–	–	–	15	0	12	–	0	M
T3	–	–	–	25	12	0	–	0	M
Kereslet	M	M	M	62+M	83+M	39+M	91	9	M

Az így kapott standard szállítási feladat adattábláját a 3.2. táblázat tartalmazza. Ha megnézzük az új táblázatot, azt láthatjuk, hogy egyrészt van egy fiktív diszkontunk 9 kereslettel és csupa nulla szállítási költséggel, ami a kiegyensúlyozás eredménye. Másrészt, minden pontra az önmagába szállítás díja 0. Az előbbi modellnél ez mindegy lenne, hiszen ennek a szállításnak nincs értelme, de most ez fontos, hogy ennyi legyen. Ennek az az oka, hogy ha szeretnénk az új utak nélküli megoldást visszakapni, ahol ugye nincs szállítás raktárak között, sem pedig diszkontok között, akkor a hozzáadott M kínálatot, illetve keresletet csak úgy tudjuk kielégíteni, ha önmagába vezető éleken elégítjük ki.

4. fejezet

Az Excel Solver

Az egyszerűbb lineáris programozási feladatok megoldásához használhatjuk az Excel táblázatkezelő Solver bővítményét. A Solver használatát egy példán keresztül mutatjuk be, majd megmutatjuk, hogy hogyan értelmezzük a Solver jelentéseit.

Tekintsük a 2.1. példa gyártástervezési feladatát. A feladatban megadott adatokat a 4.1. táblázatban újra megadjuk a könnyebbség kedvéért.

4.1. táblázat. Adatok az asztalosműhely gyártási feladatához

	Termékek				Felhasz-
Erőforrások	szék	asztal	pad	polc	nálható
famegmunkálás (óra)	1,5	1	1,5	0,3	40
bútorösszerakás (óra)	1	0,5	1,5	0,2	40
felületkezelés (óra)	1	1	1,5	0,3	40
faanyag (m^3)	0,05	0,1	0,1	0,05	2
csavar (db)	0	4	10	0	72
ragasztó (liter)	0,5	0,2	0,3	0,3	10
festék (liter)	0,5	0,5	0,7	0,4	15
Ár (ezer forint)	5	10	20	3,5	

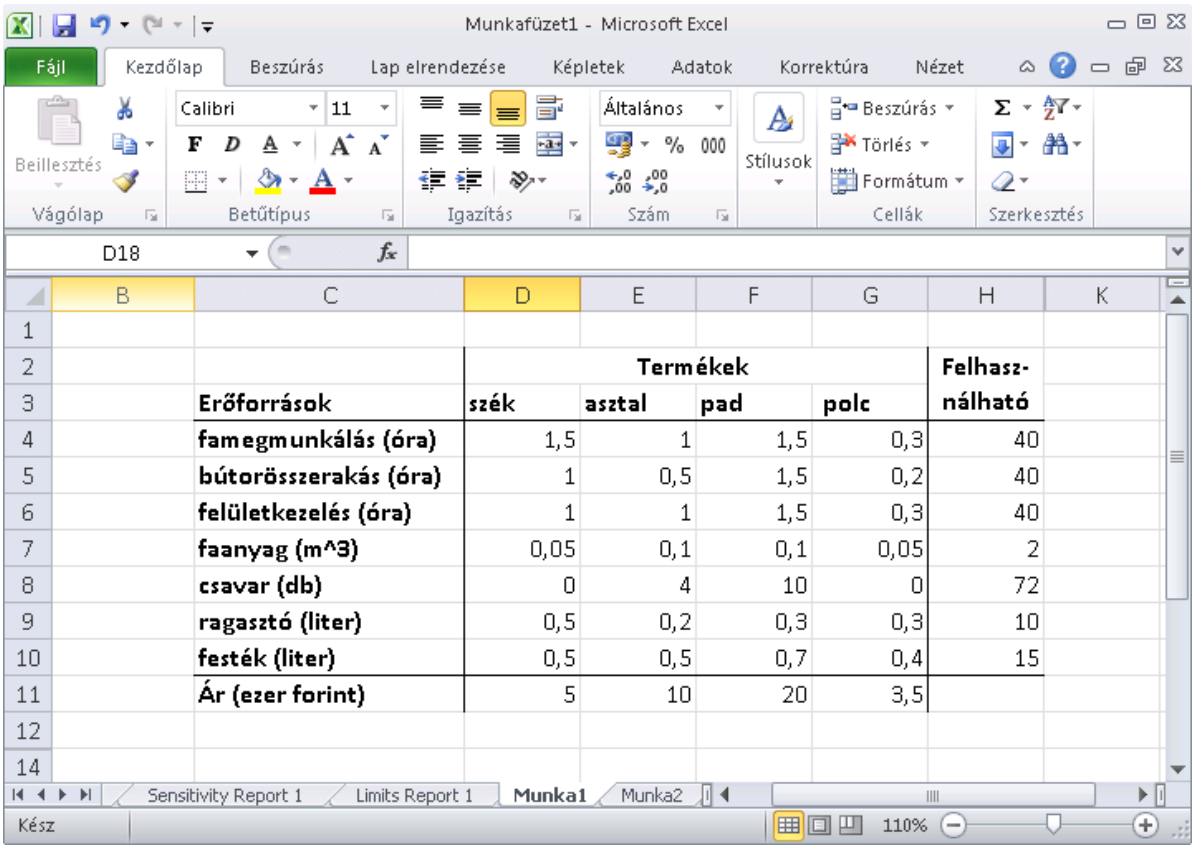
A felállított lineáris programozási modell a következő:

$$\begin{aligned} 1,5x_1 + \quad x_2 + 1,5x_3 + \; 0,3x_4 &\leq 40 \\ x_1 + 0,5x_2 + 1,5x_3 + \; 0,2x_4 &\leq 40 \\ x_1 + \quad x_2 + 1,5x_3 + \; 0,3x_4 &\leq 40 \end{aligned}$$

$$
\begin{array}{rl}
0,05x_1 + 0,1x_2 + 0,1x_3 + 0,05x_4 \leq & 2 \\
4x_2 + \ 10x_3 \leq & 72 \\
0,5x_1 + 0,2x_2 + 0,3x_3 + \ 0,3x_4 \leq & 10 \\
0,5x_1 + 0,5x_2 + 0,7x_3 + \ 0,4x_4 \leq & 15 \\
x_i \geq & 0, \forall\, i \\
\hline
\max 5x_1 + \ 10x_2 + \ 20x_3 + \ 3,5x_4 &
\end{array}
$$

4.1. Az Excel táblázat elkészítése

A feladat Excel solverrel történő megoldásához elegendő a fenti modell együttható-mátrixának a beírása a táblába, mint ahogy az a 4.1. táblázatban adott.

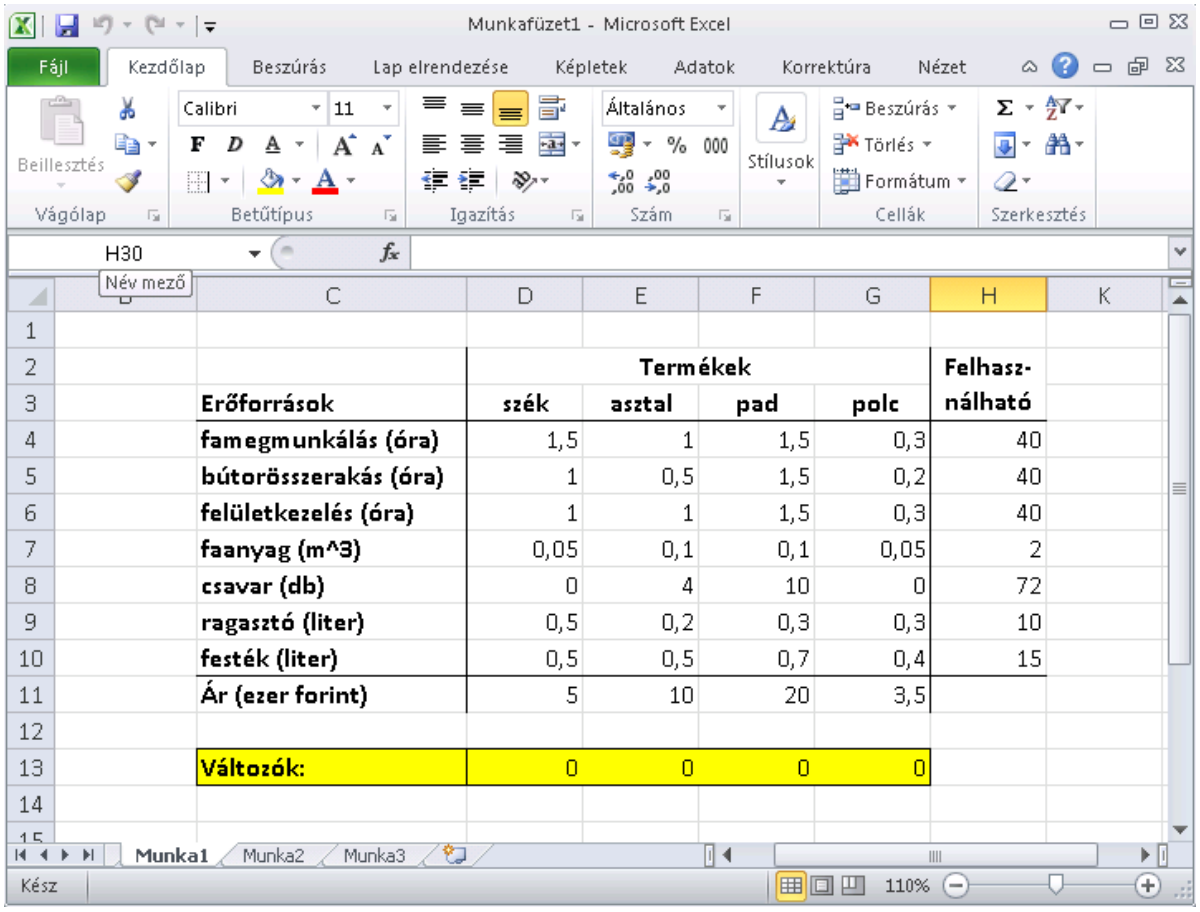


	B	C	D	E	F	G	H	K
1								
2			Termékek				Felhasz-	
3		Erőforrások	szék	asztal	pad	polc	nálható	
4		famegmunkálás (óra)	1,5	1	1,5	0,3	40	
5		bútorösszerakás (óra)	1	0,5	1,5	0,2	40	
6		felületkezelés (óra)	1	1	1,5	0,3	40	
7		faanyag (m^3)	0,05	0,1	0,1	0,05	2	
8		csavar (db)	0	4	10	0	72	
9		ragasztó (liter)	0,5	0,2	0,3	0,3	10	
10		festék (liter)	0,5	0,5	0,7	0,4	15	
11		Ár (ezer forint)	5	10	20	3,5		
12								
14								

4.1. ábra. Adatok felvétele az Excel táblázatába.

Ezután a táblázatban felvesszük a változóinkat, vagyis kijelöljük a nekik megfelelő cellákat, jelen esetben a $D13 - G13$ cellák lesznek ezek. Ezen cellák értékeit változtatja

majd meg a solver a megoldás során. Adhatunk kezdőértékeket is, például 0-kat, ez nem befolyásolja a megoldás menetét, viszont javítja az áttekinthetőséget.



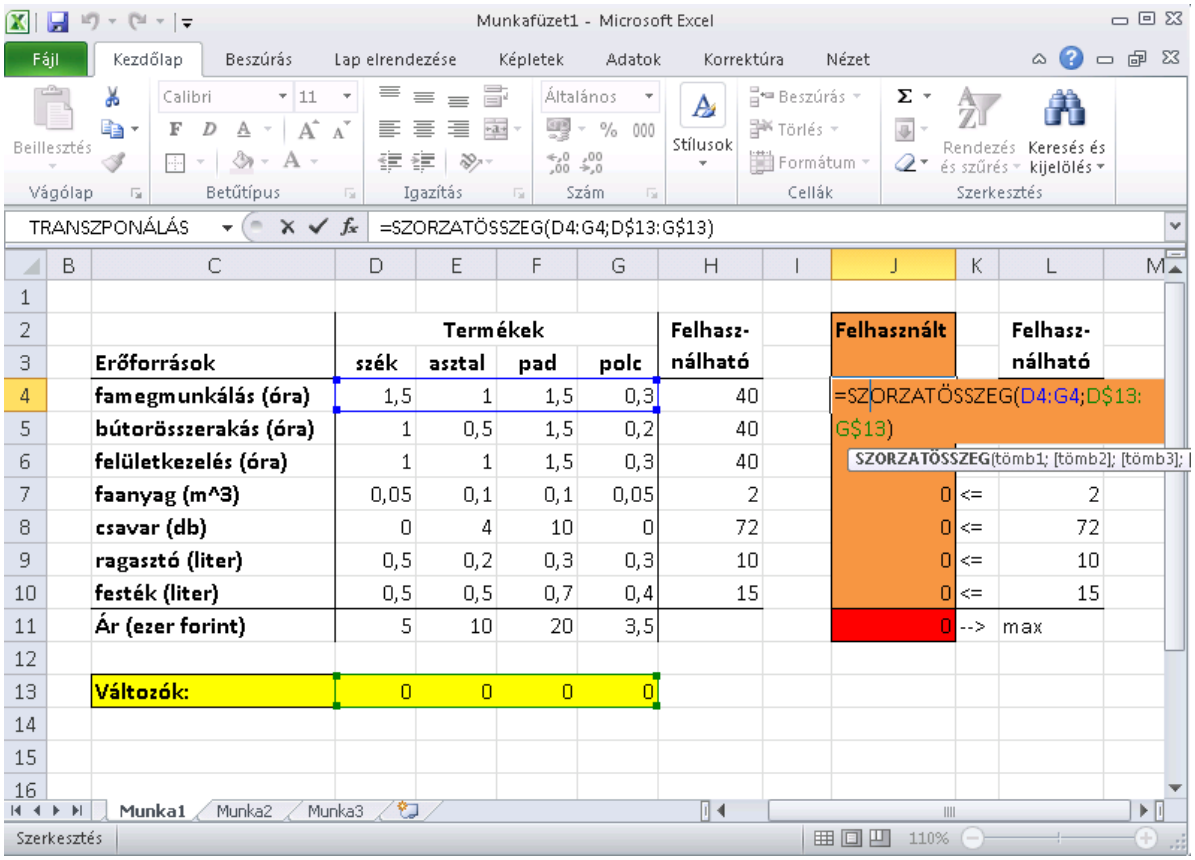
	C	D	E	F	G	H	K
1							
2		Termékek				Felhasz-	
3	Erőforrások	szék	asztal	pad	polc	nálható	
4	famegmunkálás (óra)	1,5	1	1,5	0,3	40	
5	bútorösszerakás (óra)	1	0,5	1,5	0,2	40	
6	felületkezelés (óra)	1	1	1,5	0,3	40	
7	faanyag (m^3)	0,05	0,1	0,1	0,05	2	
8	csavar (db)	0	4	10	0	72	
9	ragasztó (liter)	0,5	0,2	0,3	0,3	10	
10	festék (liter)	0,5	0,5	0,7	0,4	15	
11	Ár (ezer forint)	5	10	20	3,5		
12							
13	Változók:	0	0	0	0		
14							

4.2. ábra. Változók felvétele az Excel táblázatába.

A feltételek megadásához ki kell számolnunk a korlátozó feltételek baloldalának értékét az adott változóértékek mellett. Esetünkben ezeket a J4-J10 cellákba tettük, és a SZORZATÖSSZEG() beépített függvénnyel számoltuk ki a 4.3. ábrán látható két kijelölt vektorra: D4:G4 az együtthatóvektor, D13:G13 a változóvektor. Az ábrán látható képletben észrevehetjük, hogy a D13:G13 vektorban $ jelek vannak a sorszám, azaz a 13 előtt. Ennek az a jelentése, hogy fixáljuk le az adott információt, jelen esetben a 13-as sort a változóvektorra.

Ez akkor érdekes, ha a képletet átmásoljuk egy másik cellába. Ha $ jel nélküli cellahivatkozást másolunk, akkor a cellahivatkozás pontosan annyival tolódik el, mint amennyivel a másolt cella. Például, ha az E10 cellahivatkozást tartalmazó képletet eggyel jobbra, és eggyel feljebb másoljuk, mondjuk a G15 cellából a H14-be, akkor a H14-ben az E10 cellahivatkozás F9-re változik. Ha E10 helyett $E10 szerepelt, akkor

H14-ben $E9 lesz, mivel az oszlopot fixáltuk, ha viszont E$10 volt, akkor F$10 lesz. Könnyű kitalálni, hogy a $E$10 cellahivatkozás bárhova másolva sem változik. A mélyebb megértés érdekében érdemes eljátszani ezzel a lehetőséggel.



	B	C	D	E	F	G	H	I	J	K	L	M
1												
2			Termékek				Felhasz-		Felhasznált		Felhasz-	
3		Erőforrások	szék	asztal	pad	polc	nálható				nálható	
4		famegmunkálás (óra)	1,5	1	1,5	0,3	40		=SZORZATÖSSZEG(D4:G4;D$13:			
5		bútorösszerakás (óra)	1	0,5	1,5	0,2	40		G$13)			
6		felületkezelés (óra)	1	1	1,5	0,3	40					
7		faanyag (m^3)	0,05	0,1	0,1	0,05	2		0	<=	2	
8		csavar (db)	0	4	10	0	72		0	<=	72	
9		ragasztó (liter)	0,5	0,2	0,3	0,3	10		0	<=	10	
10		festék (liter)	0,5	0,5	0,7	0,4	15		0	<=	15	
11		Ár (ezer forint)	5	10	20	3,5			0	-->	max	
12												
13		Változók:	0	0	0	0						
14												
15												
16												

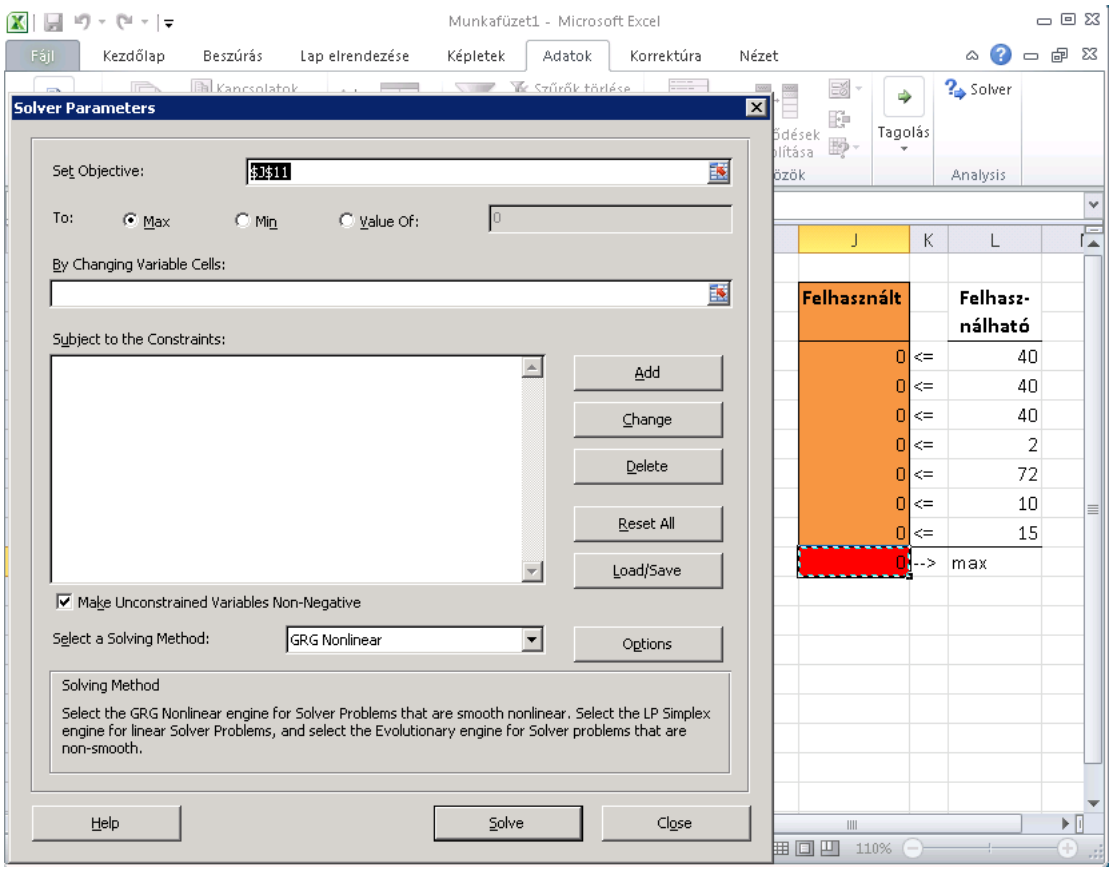
4.3. ábra. Feltételek jobboldalainak, és a célfüggvény kiszámítása Excel táblázatban.

Visszatérve a feltételek baloldalának számítására, ezzel a trükkel elegendő a famegmunkálás feltétel baloldalának képletszerű kiszámítása a fenti módon, mert a dollár jel segítségével a képlet másolása azt eredményezi, hogy amíg az első vektor mindig a megfelelő együtthatóvektorra változik, addig a változóvektor fix marad. Természetesen a $D$13:$G$13 megadás is jó eredményt ad.

A célfüggvény képlete pontosan így számítható, csak persze az ár vektorral, ami a fenti másolásával ugyanígy elérhető. Ezzel a modell táblázatba írható részével meg is vagyunk. Most kell beállítanunk a Solvert, amit a következő alfejezetben tárgyalunk.

4.2. Az Excel Solver kezelése

Ha a Solver telepítve van akkor az Adatok fül utolsó ikonjaként megjelenik, és erre kattintva tudunk elindítani. A megjelenő ablak a 4.4. ábrán látható. Ha nem jelenik meg

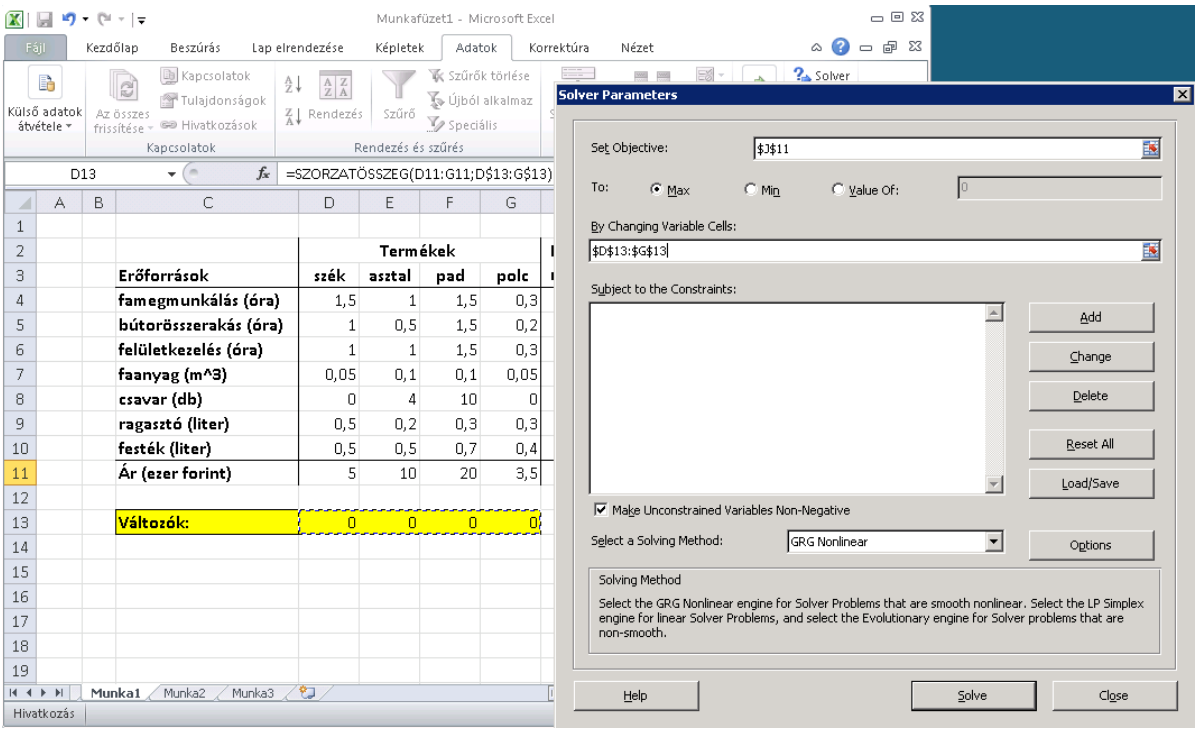


4.4. ábra. Excel Solverben a célfüggvény beállítása.

a Solver ikon az Adatok fül végén, akkor azt telepítenünk kell, ekkor a függelékben szereplő telepítési útmutatás szerint járjunk el.

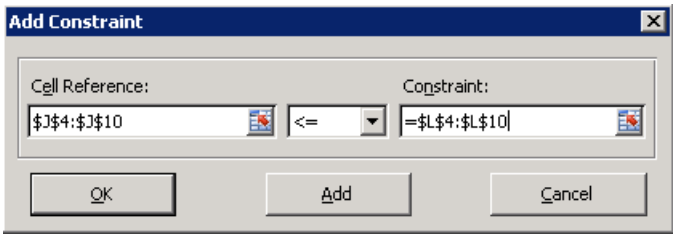
A célcella vagy jelen esetben „Set Objective” mezőbe kell a célfüggvény cellahivatkozását tennünk, jelen esetben a J11-et, ehhez az egér segítségével kiválasztjuk a megfelelő cellát. A célfüggvény jellege, ami az angol verzióban egyszerűen „To” itt már helyesen van kitöltve (max). Az érték, vagy „Value of” opció arra szolgál, ha optimalizálás helyett egy adott célfüggvényértékhez keresünk lehetséges megoldást.

A módosuló celláknál (By Changing Variable Cells) a változók celláira kell hivatkozni, ezt a 4.5. ábrán mutatjuk. Ha a változóink nem egy tartományban vannak, akkor a Ctrl billentyű segítségével választhatunk ki hozzá másik tartományt, vagy pontosvesszővel elválasztva adhatunk meg többet a sorban.



4.5. ábra. Excel Solverben a változó cellák beállítása.

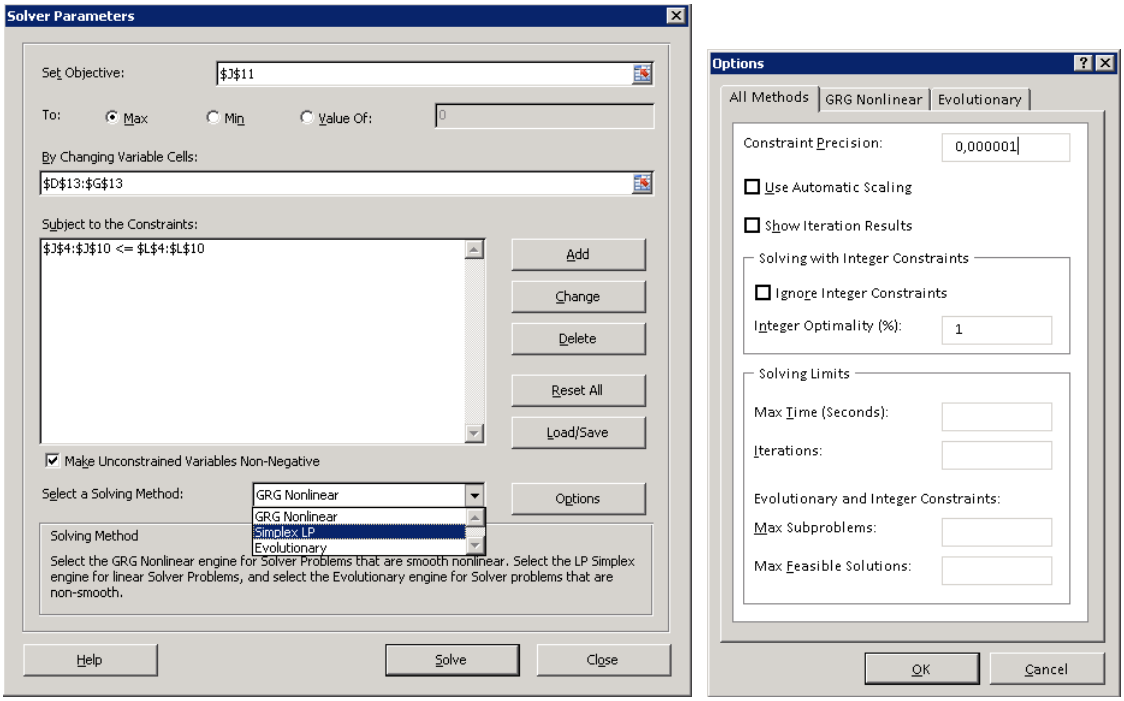
A korlátozó feltételek (Subject to the Constraints) hozzáadásánál (Add) egy új ablakban kell megjelölni a feltétel baloldalának illetve jobboldalának megfelelő cellát vagy cellákat, illetve a két oldal közti relációt. Jelen esetben a 4.6. ábrán látható a megfelelő kitöltés, amivel az összes erőforrásfeltételt megadtuk. Ha további feltételek hozzáadása szükséges, akkor a hozzáadás (Add), egyébként az OK gombra kattintsunk. Fontos megjegyezni, hogy itt adhatunk meg a változóinkhoz egészértékű feltételeket, ha a relációk közül az int (integer – egész) vagy a bin (binary – bináris) mezőt választjuk.



4.6. ábra. Excel Solverben a feltételek hozzáadása.

A feltételek bevitele után meg kell adnunk, hogy a változóink nemnegativitási feltételt teljesítenek-e (Make Unconstrained Variables Non-negative), ez az alapbeállítás

ebben a verzióban, ami nekünk pont jó, ezért nem is adtuk hozzá ezeket a feltételeket külön. Szintén itt kell eldöntenünk, hogy a feladatot milyen megoldóval szeretnénk megoldani. Lineáris programozási feladatokat, a Szimplex módszerrel (Simplex LP) oldunk meg, mint ezt is. Ha adtunk meg egészértékű feltételt, akkor is a Simplex LP megoldót válasszuk, ebben az esetben viszont a Korlátozás és szétválasztás módszerén belül fogjuk használni az egyes részproblémák megoldására a szimplex módszert.



(a) Feltételek hozzáadása. (b) Módszerek beállításai.

4.7. ábra. Excel Solver beállításai.

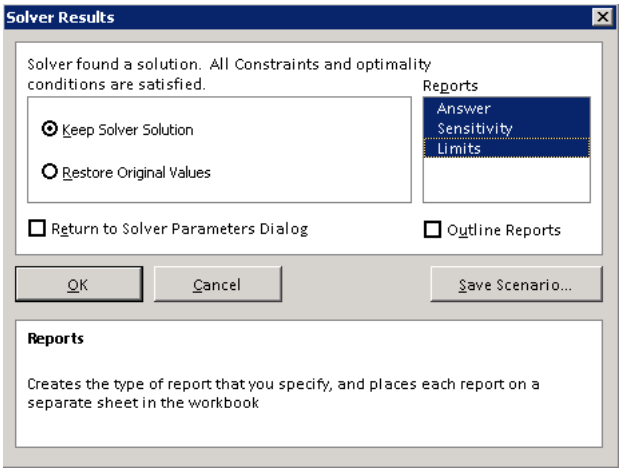
Az opciók (Options) ablakban a módszer paramétereit tudjuk beállítani. A 4.7(b). ábrán az első állítható opció a feltételek pontossága (Constraint Precision) ami azt adja meg, hogy a feltétel legfeljebb mekkora értékkel sérthető meg. Az automatikus skálázás (Use Automatic Scaling) opcióval a változók értékeit úgy skálázza a módszer, hogy a numerikus stabilitás a legjobb legyen. A lépésenkénti kijelzés (Show Iteration Results) akkor hasznos, ha nyomon akarjuk követni a módszer lépéseit, akár ha a szimplex módszer kézzel számolt iterációt ellenőrizzük. Az ezt követő blokk csak egészértékű változók esetén érdekes, itt állíthatjuk be egyrészt az egész feltételek relaxációját (Ignore Integer Constraints) illetve a megkövetelt pontosságot (Integer Optimality), ami a legjobb egészmegoldás és a legjobb korlát közötti különbség százalékos arányban.

Végül a megállási feltételek (Solving Limits) blokk következik, ahol a maximális futási időt (Max Time) adhatjuk meg másodpercekben, a maximális iterációs lépésszá-

mot (Iterations), illetve az evolúciós és egészértékű problémák esetén a részproblémák maximális számát (Max Subproblems) illetve a megengedett megoldások maximális számát (Max Feasible Solutions).

Ezek után a beállításokat jóváhagyva megnyomhatjuk a „Megoldás" gombot (Solve). Az előugró ablakban lévő információkat figyelmesen olvassuk el. Négyféle választ kaphatunk (például mint a 4.8. ábrán):

- „Solver found a solution. All Constraints and optimality conditions are satisfied." Vagyis a Solver talált egy megoldást, amelyre minden feltétel és optimalitási kritérium teljesül.
- „Solver could not find a feasible solution." Ekkor a Solver nem talált egyetlen lehetséges (megengedett) megoldást, azaz a feltételek között ellentmondás van.
- „The Objective Cell values do not converge." Ebben az esetben nem korlátos a feladat, így nem konvergál a módszer. Ilyenkor is a feltételek között kell körülnézni, mert hiányzik (vagy rosszul adott) egy olyan feltétel, ami a célfüggvényt korlátozná.

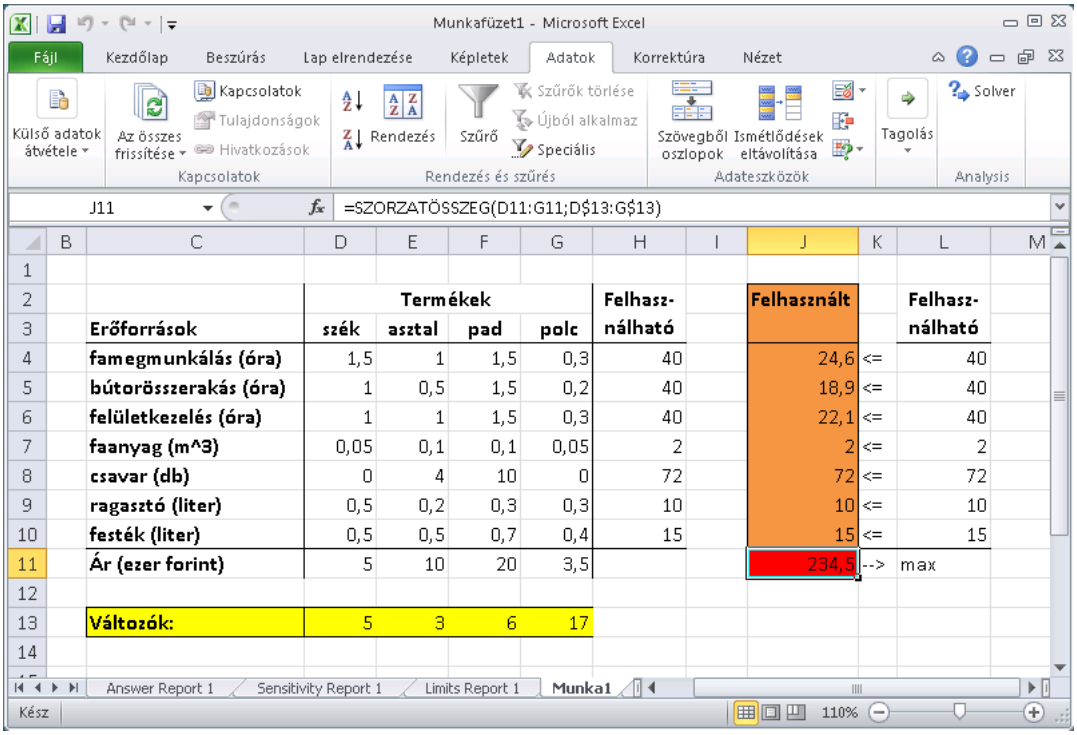


4.8. ábra. A Solver optimális megoldást talált.

- „The maximum /iteration limit/time limit/number of subproblem/number of integer solution/ was reached; continue anway?" A megjelölt megállási feltétel miatt megállt az algoritmus, de folytathatjuk tovább a módszert, vagy megállhatunk az aktuális eredménnyel. Ez a válasz persze csak akkor fordulhat elő ha valamilyen megállási feltételt beállítunk, alapértelmezésben csak a fenti három válasz lehetséges.

A válasznak megfelelően eldönthetjük, hogy megtartjuk a Solver megoldását (Keep Solver Solution), vagy visszatérünk az eredeti értékekhez (Restore Original Values), illetve hogy a Solver beállítási ablakot kérjük-e vissza (Return to Solver Parameters Dialog). Az ablak jobboldalán a Reports (jelentések) blokkban pedig lehetőségünk van a megoldás részletesebb elemzését is kérni. Itt mindet bejelöltük.

Az „OK" gomb megnyomása után a Solver a bejelölésnek megfelelően három munkalapot generál, Answer Report (Eredményjelentés), Sensitivity Report (Érzékenységjelentés) és Limits Report (Határok jelentés). Az eddigi táblázatunkban (4.9. ábra) pedig láthatjuk a feladat optimális megoldását: 5 szék, 3 asztal, 6 pad és 17 polc gyártásával kapjuk a maximális bevételt, ami 234 500 Ft.



4.9. ábra. A Solver által adott optimális megoldás.

4.3. A Solver jelentései

Most pedig végigmegyünk a három jelentésen, és megmutatjuk a kapott adatok értelmezését.

Microsoft Excel 14.0 Answer Report
Worksheet: [Munkafüzet1]Munka1
Report Created: 2013.07.25. 15:13:34
Result: Solver found a solution. All Constraints and optimality conditions are satisfied.
Solver Engine
Engine: Simplex LP
Solution Time: 0,016 Seconds.
Iterations: 4 Subproblems: 0
Solver Options
Max Time Unlimited, Iterations Unlimited, Precision 0,000001
Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%, Assume NonNegative

Objective Cell (Max)

Cell	Name	Original Value	Final Value
$J$11	Ár (ezer forint) Felhasznált	0	234,5

Variable Cells

Cell	Name	Original Value	Final Value	Integer
$D$13	Változók: szék	0	5	Contin
$E$13	Változók: asztal	0	3	Contin
$F$13	Változók: pad	0	6	Contin
$G$13	Változók: polc	0	17	Contin

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$J$4	famegmunkálás (óra) Felhasznált	24,6	$J$4<=$L$4	Not Binding	15,4
$J$5	bútorösszerakás (óra) Felhasznált	18,9	$J$5<=$L$5	Not Binding	21,1
$J$6	felületkezelés (óra) Felhasznált	22,1	$J$6<=$L$6	Not Binding	17,9
$J$7	faanyag (m^3) Felhasznált	2	$J$7<=$L$7	Binding	0
$J$8	csavar (db) Felhasznált	72	$J$8<=$L$8	Binding	0
$J$9	ragasztó (liter) Felhasznált	10	$J$9<=$L$9	Binding	0
$J$10	festék (liter) Felhasznált	15	$J$10<=$L$10	Binding	0

4.10. ábra. Eredmény jelentés.

Eredményjelentés

A 4.10. ábrán láthatjuk az eredményjelentés által adott információkat. Leolvashatjuk mi volt a használt módszer, a számítási időt (0,016s) és az iterációk számát is. Egészértékű problémák esetén a Subproblems adja meg a megoldott részproblémák számát. A Solver Options szekcióban a beállított opciók kerülnek felsorolásra.

A táblázatok a feladat három főrészéhez kapcsolódnak: Célfüggvény (Objective cell), változók (Variable cells) és Feltételek (Constraints). Mindegyik esetben az első oszlopban a megfelelő cellahivatkozás, az Excel által hozzárendelt név, majd az eredeti és végső érték (original, final value) van megadva. A célfüggvény táblázata ezzel nem sok újdonságot tartalmaz, megadja az optimális célfüggvény értékét. A változók táblázatában az értékeken kívül azt is leolvashatjuk, hogy melyik változónk volt folytonos (Continuous), egész (Integer), vagy bináris (Binary). Ebben a példában nem adtunk meg egészértékű feltételeket, így minden változónk folytonos. Itt jegyezzük meg, hogy ellenkező esetben a megoldásról nem készül érzékenységjelentés.

A Feltételek táblázatban megtaláljuk a feltételek formuláját is, illetve a státuszukat is az optimális megoldás mellett. A Binding (magyar verzióban Éppen) tulajdonságú feltételek esetén egyenlőség teljesül, míg a Not Binding (magyar verzióban Bőven) feltételeknél a szigorú kisebb vagy nagyobb reláció teljesül. Az utolsó (Slack) oszlopban az is látható, hogy a rendelkezésre álló erőforrásokból mennyi marad meg az optimális megoldás mellett. Itt a Slack, a maradékváltozó értéke tulajdonképpen.

Érzékenységjelentés

Az érzékenységjelentés két táblázatot tartalmaz, hasonlóan az eredmény jelentéshez, de itt több információt tudunk leolvasni az egyes értékekből. Nézzük előbb a Változók táblázatát. A szokásos cellahivatkozás és név után találjuk a változók optimális értékét, majd a „Reduced Cost" oszlopban a redukált költséget, ami azt mondja meg, hogy egy az optimális termelésben nem szereplő árú esetén mennyivel kell redukálni a költségét (vagy emelni az árát) az adott terméknek, hogy bekerüljön a bázisba, azaz termeljünk belőle. A mi esetünkben most mindegyik terméket gyártjuk, de esetleg megnézhetjük mi történik, ha az egyik termék árát drasztikusan lecsökkentjük, és újrageneráljuk ezt a jelentést. A további oszlopok hasonló információt adnak. Az „Objective Coefficient" tartalmazza a célfüggvényegyütthatókat, jelen esetben az árakat, és mellette ezek megengedett növekedését illetve csökkenését (Allowable Increase és Allowable Decrease), hogy az optimális megoldás ne változzon (a célfüggvényérték természetesen változhat). Fontos megjegyezni, hogy ezek az értékek csak az adott együttható változásainak az érzékenységét adják meg, vagyis ha minden más változatlan marad. Több együttható együttes változását nem tudjuk így vizsgálni. Nézzünk erre pár példát.

- Ha a szék árát 5000 Ft-ról több mint 71,5 Ft-tal növeljük, akkor már más megoldást kapunk (növeljük a székek számát valami más bútor kárára), hasonlóan, ha

legalább 654 Ft-tal csökkentjük az árát, akkor vélhetően kevesebbet fogunk belőle gyártani.

- A polc 3500 Ft-os árát elég csak 40 forinttal csökkenteni ahhoz, hogy csökkenjen a mennyisége az optimális gyártásban, viszont ha 400 Ft-tal nagyobb árat kérünk érte, akkor többet kell készítenünk belőle valamely más bútor kárára.

Microsoft Excel 14.0 Sensitivity Report
Worksheet: [Munkafüzet1]Munka1
Report Created: 2013.07.25. 15:13:34

Variable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
$D$13	Változók: szék	5	0	5	0,071428571	0,653846154
$E$13	Változók: asztal	3	0	10	0,1	0,5
$F$13	Változók: pad	6	0	20	1,25	0,25
$G$13	Változók: polc	17	0	3,5	0,357142857	0,038461538

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
$J$4	famegmunkálás (óra) Felhasznált	24,6	0	40	1E+30	15,4
$J$5	bútorösszerakás (óra) Felhasznált	18,9	0	40	1E+30	21,1
$J$6	felületkezelés (óra) Felhasznált	22,1	0	40	1E+30	17,9
$J$7	faanyag (m^3) Felhasznált	2	20,83333333	2	0,291428571	0,072
$J$8	csavar (db) Felhasznált	72	1,520833333	72	37,09090909	24
$J$9	ragasztó (liter) Felhasznált	10	7,083333333	10	1,073684211	0,461538462
$J$10	festék (liter) Felhasznált	15	0,833333333	15	0,36	0,784615385

4.11. ábra. Érzékenységjelentés.

A fenti megállapításokat az összes változóra megtehetjük, de mindig érdemes kiragadni azokat, amelyek vagy nagyon érzékenyek (kis változtatással változik az eredmény), vagy nagyon stabilak (nem érzékenyek a kis változásokra). Ezeket a megállapításokat ki is próbálhatjuk: nézzük meg, hogy változik a megoldás, ha egyik-másik árat megváltoztatjuk. Megfigyelhetjük, hogy ha valamely áruból nem termelünk, akkor a redukált költsége megegyezik a megengedett növekedés mínuszegyszeresével, ami teljesen összhangban van az eddig elmondottakkal, illetve a megengedett csökkenés 1E+30, ami a 10^{30}-t jelöli és arra utal, hogy bármennyivel csökkenthető innen a célfügg-

vényegyüttható, a megoldás nem fog változni. Ez persze logikus, hiszen ha az aktuális áron nem volt értelme termelni, akkor kisebb árért sem lesz értelme.

A Feltételek táblázata nagyon hasonló módon értelmezhető, és tulajdonképpen tekinthetjük úgy, mint a duális feladat változóihoz tartozó „Változó" táblázat. A változók táblázatához képest a Redukált költség helyett itt a Shadow Price azaz árnyékár van megadva, ami azt mondja meg, hogy egységnyi erőforrást maximum milyen áron érdemes vennie a vállalatnak további profit eléréséhez. Természetesen azoknál az erőforrásoknál, ahol nem használtuk ki az összes rendelkezésre álló mennyiséget, ez az ár nulla, hiszen nem éri meg többet venni a megmaradtak mellé. Az utolsó három oszlop hasonlóan értelmezhető, mint a változók esetén. A feltételek jobboldala (Constraints R.H. Side) megadja az aktuálisan rendelkezésre álló mennyiségeket, míg az Allowable Increase és Allowable Decrease ezek megengedett növekedését illetve csökkenését mutatják az optimális bázis változatlansága mellett (a bázisváltozók értéke változhat). Itt is csak egyetlenegy együttható változtatása mellett igazak az értékek. Nézzünk erre is pár példát:

- Láthatjuk, hogy a munkaórák minden esetben kihasználatlanok, így a megengedett növekedés végtelen, bármennyivel is növeljük a munkaórák számát, nem változik a megoldás, így a bázis sem. Azt is leolvashatjuk, hogy a bútorösszerakásból marad meg a legtöbb óránk, és ha a munkásaink több feladathoz is értenek, akkor egy munkást akár el is bocsájthatunk a jelenlegi termelési terv mellett (összesen 54,4 kihasználatlan munkaóránk van).

- Alapanyagok tekintetében nincs feleslegünk viszont. Az egyik legérzékenyebb alapanyagunk a fa, hiszen a raktáron lévő fa több $0{,}072\text{m}^3$-el való csökkenése már új optimális termelési tervet írna elő. A másik érzékeny alapanyag a festék, ahol több mint 3,6dl növekedés eredményez új bázismegoldást.

- A legstabilabb alapanyagunk a csavar, hiszen ennek sem kismértékű csökkenése sem kismértékű növekedése nem ad változást a megoldás bázisában.

Hasonlóan a változókra tett megállapításokhoz, itt is érdemes pár együtthatóra kipróbálni, mi lenne az optimális megoldás, ha a megadott értékeknél nagyobb, vagy éppen kevesebb lenne az erőforrásból rendelkezésre álló mennyiség. Az így kapott eredmények a vállalat vezetőinek fontos információk, hiszen ők ismerik az alapanyagok beszerzési árait, és láthatják a lehetőségeket és veszélyeket az egyes változtatásokban.

Határok jelentés

Az utolsó jelentésünk a változók határait adja meg. Vagyis az adott optimális megoldásból kiindulva, egy-egy változót milyen határok közt tudunk mozgatni úgy, hogy a korlátozó feltételek továbbra is teljesüljenek, és a határokon felvett változóértékek

mellett mekkora ekkor a célfüggvény értéke. Mivel a termékekre nem volt adott minimális termelési követelmény, így minden esetben a nulla az alsó határ (Lower Limit), és megkapjuk egyenként azt a bevételt, ami egy-egy termék nem gyártása eredményez (az összes többi változatlansága mellett). Például, ha minimális számú padot gyártunk, azaz nem gyártunk egyet sem, akkor a bevételünk 114 500 Ft, míg a maximális 6 polc

Microsoft Excel 14.0 Limits Report
Worksheet: [Munkafüzet1]Munka1
Report Created: 2013.07.25. 15:13:34

		Objective	
Cell	Name	Value	
$J$11	Ár (ezer forint) Felhasznált	234,5	

		Variable		Lower	Objective	Upper	Objective
Cell	Name	Value		Limit	Result	Limit	Result
$D$13	Változók: szék	5		0	209,5	5	234,5
$E$13	Változók: asztal	3		0	204,5	3	234,5
$F$13	Változók: pad	6		0	114,5	6	234,5
$G$13	Változók: polc	17		0	175	17	234,5

4.12. ábra. Határok jelentés.

gyártása esetén 234 500 Ft. Ha egy feltételünk egyenletként definiált, akkor a változó értéket megváltoztatva ez a feltétel már nem teljesülne. Emiatt a változókra kapott alsó és felső határok ilyen esetben egybeesnek.

5. fejezet

Az AMPL leíró nyelv

Az AMPL (**A** **M**athematical **P**rograming **L**anguage), mint ahogy azt a neve is mutatja matematikai programozási feladatok leírására szolgál. Ahogy a továbbiakban látni fogjuk, modelljeink leírása meglehetősen kézenfekvő ezen a nyelven.

Mint eddig is, példákon keresztül fogjuk bevezetni az olvasót az AMPL rejtelmeibe.

5.1. Példa. Feltesszük, hogy van n lehetséges termék amit előállíthatunk, és b óránk az előállításra. Minden termékre adott az egységnyi nyereség, a maximum előállítható mennyiség és hogy mennyi terméket állítunk elő egy óra alatt. Maximalizáljuk a profitot!

Megoldás: Legyenek a változóink a termékek előállítandó mennyiségei, azaz $x_1, \ldots, x_n$. Mivel a feladatban nem szerepelnek konkrét adatok, így paraméterekkel írjuk fel a modellt. Jelöljük a termékek egységnyi nyereségét a $p_1, \ldots, p_n$ paramétervektorral. Így a célfüggvényünk

$$\max \sum_{i=1}^{n} p_i x_i.$$

Jelölje a maximum előállítható mennyiségeket $m_1, \ldots, m_n$, míg az egy óra alatt előállítható mennyiségeket $o_1, \ldots, o_n$. A maximumra vonatkozó feltételek felírása egyszerű,

$$x_i \leq m_i, \ \forall\, i \in \{1, \ldots, n\}.$$

Az óraszámra vonatkozó feltétel kicsit trükkösebb, itt fontos észrevenni, hogy a megadott paramétereknek a reciproka igazán érdekes számunkra, hiszen $1/o_i$ adja meg az egységnyi i. termék előállításához szükséges időt. Vagyis a termeléssel töltött idő $\sum_{i=1}^{n} x_i/o_i$, amire adott korlátunk b. Összefoglalva a modell a következő.

$$\sum_{i=1}^{n} \frac{x_i}{o_i} \leq b$$

$$0 \leq x_i \leq m_i, \forall\, i \in \{1, \ldots, n\}$$
$$\max \sum_{i=1}^{n} p_i x_i.$$

A következő alfejezetben megadjuk ennek a modellnek az AMPL-ben felírt változatát, és egy példa adatra meg is oldjuk.

5.1. AMPL alapok

A fentiekben tárgyalt egyszerű modellnek az AMPL kódját a következőképpen adhatjuk meg. Először definiáljuk az index halmazokat, jelen esetben egy ilyen halmazunk van, a termékek halmaza, nevezzük most ezt `PROD`-nak. A modell fájlban ez egyszerűen a `set PROD;` sor lesz, hiszen itt nem szükséges még megadni, hogy pontosan milyen és mennyi termékünk lesz. A halmaz definíciók után megadjuk a szükséges paramétereket, lehetőség szerint beszédes nevekkel, hogy a kód többi része is olvasható legyen. Például legyen `oraalatt` az egy óra alatt előállítható termékek mennyisége. Mivel ez minden termékre különböző, egy vektorként definiáljuk, azaz `param oraalatt {j in PROD};` a megfelelő parancs. Hasonlóan adhatjuk meg a többi paramétert is.

Ezek után következik a változók, majd a modell megadása. A változókat a `var` kulcsszó után adhatjuk meg hasonlóan a paraméterekhez, és itt már néhány feltételt is megfogalmazhatunk rájuk. A példánkban ez a `var X {j in PROD} >=0;` parancsot jelenti, ahol megadtuk, hogy a változóink nemnegatívak.

A `maximize` kulcsszó adja meg, hogy a célfüggvény definíciója következik. A kulcsszó után nevet adunk a célfüggvénynek, majd jöhet a képlet. Példánkban ez a `maximize Profit: sum {j in PROD} profit[j] * X[j];` parancs. Itt könnyen kitalálható a `sum` függvény jelentése.

A feltételek megadása nagyon hasonló lesz. A feltételeket a `subject to` kulcsszóval adjuk meg, amit szintén egy név követ, majd kettőspont után a feltétel képlete. Például a maximális óraszám feltételt a `subject to Orak: sum {j in PROD} (1/oraalatt[j]) * X[j] <= b;` parancs jelenti. A teljes modell a 5.1. kódlistában látható.

A fenti kód ugyan elég jól olvasható, azért térjünk ki néhány fontos szintaktikai részletre. Minden parancs `;`-re végződik, egy parancs több sorban is lehet. Az alapvető kulcsszavak, azaz `set`, `param`, `var`, `maximize`, `minimize`, `subject to` mindegyikét egy egyedi név követ, amire hivatkozhatunk a későbbiekben. Ha egyszerre több változót, vagy feltételt szeretnénk megadni, akkor kapcsoszárójelek között meg kell adnunk, hogy mely halmaz elemeire legyen definiálva az adott változó vagy feltétel.

A halmaz és paraméter deklarációknál, ha itt kerülnek definiálásra, akkor a halmaznév után `:=` -vel elválasztva adjuk meg a definíciót, például `set P1; set P2;`

5.1. Kódlista: gyartas.mod

```
set PROD; # a termékek halmaza

param oraalatt {j in PROD}; # egy óra alatt gyártott termék
param b; # maximum óraszám
param profit {j in PROD}; # termékenkénti nyereség
param maxgyart {j in PROD}; # maximálisan gyártható mennyiség

var X {j in PROD}>=0; # gyártandó mennyiségek

maximize Profit: sum {j in PROD} profit[j] * X[j];

subject to Orak: sum {j in PROD} (1/oraalatt[j]) * X[j] <= b;
subject to Hatarok{j in PROD}: X[j] <= maxgyart[j];
```

```
set P3:=P1 union P2; P4:=1..5;.
```

A változók esetén bizonyos feltételeket akár a deklarációnál is megadhatunk. Példánkban láthattuk, hogy a nemnegativitási feltétel egyszerűen megadható. Hasonlóan definiálhatunk nempozitív, egész, bináris változókat, de akár konstans korlátúakat is: `var y <=0 integer; var z binary; var w >= 100 <= 200;`

A feltételek és a célfüggvény megadásánál mindig a `:` jelet követően adjuk meg a képletet. Vegyük észre, hogy a standard alakú lineáris programozási feladatok feltételrendszere egy sorban megadható:

```
subject to feltetel{i in F}: sum{j in V} A[i,j]*x[j] = b[i];
```

A célfüggvényhez két kulcsszóból választhatunk, `maximize` vagy `minimize`, ezek a nevek önmagukért beszélnek.

A képletekben a legtöbbet használt függvény a `sum`, szintaktikájánál fontos, hogy több tag esetén zárójelezni kell a tagokat, mert ellenkező esetben csak az elsőt szummázza össze, vagyis a `sum{i in I} x[i]+y[i]` csak az x_i tagokat szummázza, és fordításnál hibát fog jelezni arra, hogy az y_i-ben az i nem definiált. A jó képletet a `sum{i in I} (x[i]+y[i])` adja meg.

Ezzel a modell megadásával meg is volnánk. A feladat egy konkrét megoldásához tegyük fel, hogy adottak az értékek a megfelelő paraméterekhez. Ezeket az értékeket az úgynevezett adatfájlban tudjuk megadni. Legyen most a két termékünk p1 és p2, ezt az adatfájlban a `set Prod:=p1 p2;` paranccsal adhatjuk meg. Paraméter vektorok megadásánál fontos, hogy mindig meg kell adni az indexet és majd a hozzátartozó értékeket, például `param profit:= p1 25 p2 30;` egy szintaktikailag jó megadás. Több vektor együttes megadása is lehetséges, ha ugyanazon halmaz felett definiáltak.

Ennek szintaktikai megoldását láthatjuk az `oraalatt` és `maxgyart` vektorokra a teljes adatfájlban, amit az 5.2. kódlistában közöltünk.

5.2. Kódlista: gyartas.dat

```
set PROD := p1 p2;

param b := 40;

param profit:=
p1 25
p2 30;

param: oraalatt maxgyart :=
p1 200 6000
p2 140 4000 ;
```

Így lényegében készen is volnánk, már csak át kell adni az információt az AMPL parancsértelmezőnek. Indítsuk el tehát a programot, ami linuxos és windowsos környezetben is parancssorból futtatható csak (ha még nincs telepítve kövessük a A.2. fejezetben leírt telepítési útmutatót). Ha sikeresen elindult a program, akkor az `ampl:` promptot adja. A fenti fileokat beolvashatjuk a `model gyartas.mod` és `data gyartas.dat` parancsokkal, amikre ha csak az `ampl:` választ kapjuk, akkor ügyesek voltunk, a parancsértelmező nem talált hibát bennük (azért sajnos még lehet benne hiba, ilyenkor a megoldó jelez vissza). Ezek után megoldható a feladat egy kiválasztott solver segítségével. Ezt egy opció beállításával megadhatjuk, például `option solver cplex;` vagyis a CPLEX megoldót használjuk. Ezek után a `solve` parancs elindítja a megoldót, ami kiír a futásról néhány információt, és általában a kapott megoldás célfüggvényértékét is. Ha kíváncsiak vagyunk a változóink értékére, akkor a `display` paranccsal ezeket kiírathatjuk, például `display X;`. Az így beírt sorok a mi esetünkben a következő eredményt adták.

```
user@pc:/home/user$ ampl
ampl: model gyartas.mod
ampl: data gyartas.dat
ampl: option solver cplex;
ampl: solve;
CPLEX 12.5.1.0: optimal solution; objective 192000
1 dual simplex iterations (0 in phase I)
ampl: display X,Profit;
X [*] :=
```

```
p1  6000
p2  1400
;

Profit = 192000

ampl:
```

Vagyis a módszer egy duál szimplex lépéssel megtalálta az optimumot, ami p1 termékből 6000, míg p2 termékből 1400 darabot jelent, és az így elérhető profit 192 000 Ft.

Az említett parancssorokat összefoglalva egy futtató (.run) fileba a megoldást egy paranccsal is megkaphatjuk `ampl gyartas.run`. A run fileunkat az 5.3. kódlistában olvashatjuk.

5.3. Kódlista: gyartas.run

```
model gyartas.mod;
data gyartas.dat;
option solver cplex;
solve;
display X, Profit;
```

5.2. AMPL haladóknak

Ezután az egyszerű példa után gyakoroljuk a tudásunkat egy nehezebb feladaton, ahol további trükköket is mutatunk.

5.2. Példa. Egy cég 3 termék gyártását tervezi januártól április végéig. A keresletet az egyes termékekre az 5.1. táblázatban adtuk meg.

A termékekhez tartozó árak, gyártási költségek, aktiválási költségek, maximum és minimum előállítható mennyiségek az 5.2. táblázatban találhatóak. Itt az aktiválási költség azt jelenti, hogy minden hónapban ennyibe kerül, ha gyártunk az adott termékből, a minimális mennyiség ehhez kapcsolódóan, az a mennyiség, amennyit le kell gyártani, ha már gyártjuk. A megadott maximális mennyiség úgy értendő, hogy ha csak azt a terméket gyártják, akkor mennyit tudnának egy nap alatt elkészíteni.

Adott továbbá, hogy januárban 23, februárban 20, márciusban 23, míg áprilisban 22 munkanap van. Mind az aktiválási költség mind a minimum mennyiség havonta értendő. Lehetőségünk van a legyártott termékek tárolására is. Ennek költsége az A1, A2, A3 termékekre rendre 3,50$, 4,00$, illetve 3,00$. Minden termék egységnyi

5.1. táblázat. Az A1-A3 termékek kereslete az 5.2. példához.

	Jan.	Febr.	Marc.	Apr.
A1	5300	1200	7400	5300
A2	4500	5400	6500	7200
A3	4400	6700	12500	13200

5.2. táblázat. Az A1-A3 termékek költségei az 5.2. példához.

Termék	A1	A2	A3
Egységár	124$	109$	115$
Aktiválási költség	150000$	150000$	100000$
Gyártási költség	73,30$	52,90$	65,40$
Maximum db/nap	500	450	550
Minimum mennyiség	20	20	16

helyet foglal, és összesen 800 egység tárolására van lehetőség. A vállalat célja, hogy maximalizáljuk a profitot!

Megoldás: Kezdjük a megoldást a matematikai modell megadásával. Ebben a feladatban már a változók megadása sem egyszerű feladat. Vegyük észre, hogy a gyártott és eladott termékek nem ugyanazok, illetve számolni kell a tárolt termékekkel is. Mint sok más esetben is, itt is elég a célfüggvény felírása ahhoz, hogy megtudjuk, milyen változókra van szükségünk.

Ha a profitot akarjuk maximalizálni, akkor a bevételt és a költségeket kell tudnunk felírni. A bevétel az az árszor az eladott mennyiség, vagyis az eladott mennyiség lesz itt a változónk. A költségek már több dologból tevődnek össze, van a gyártás költsége, ami a gyártott mennyiség szorozva a gyártási költséggel, a tárolás költsége ami ugyanígy tárolt mennyiség szorozva a tárolási költséggel, illetve az aktiválás költsége, ami az aktiválási költség szorozva az aktiváltsági állapottal, ami 0 vagy 1. Vagyis az így adott változóink: a gyártott, az eladott, a tárolt mennyiségek, illetve az aktiváltsági állapot, és ezek minden termékre és minden hónapra külön-külön kellenek.

Vezessük be a következő jelöléseket:

Paraméterek

d_{ij} kereslet az i. hónapban a j. termékre

gyk_j a j. termék gyártási költsége

tk_j a j. termék tárolási költsége

ak_j a j. termék aktiválási költsége

ar_j a j. termék ára

mm_j a j. termék minimális mennyisége

db_j a j. termék egy nap alatt gyártható maximális darabszáma

n_i az i. hónap munkanapjainak a száma

Változók

x_{ij} az i. hónapban a j. termékből gyártott mennyiség

y_{ij} az i. hónapban a j. termékből eladott mennyiség

t_{ij} az i. hónapban a j. termékből tárolt mennyiség

a_{ij} az i. hónapban a j. termék aktivált-e

Ezekkel a jelölésekkel írjuk fel először a célfüggvényt.

$$\max \sum_i \left(\sum_j ar_j y_{ij} - \sum_j gyk_j x_{ij} - \sum_j tk_j t_{ij} - \sum_j ak_j a_{ij} \right)$$

A korlátozó feltételek közül először írjuk fel az egyszerűbbeket. A legkézenfekvőbb a kereslet feltétele, hiszen minden hónapban minden termékre az eladott mennyiség nem haladhatja meg a hozzá tartozó keresletet:

$$y_{ij} \leq d_{ij} \quad \forall i, j.$$

A maximális tárolást is könnyű megadni, minden hónapra a raktárkapacitást nem haladhatja meg az összes tárolt termék mennyisége:

$$\sum_j t_{ij} \leq 800 \quad \forall i.$$

Még adjuk meg a nemnegativitási feltételeket, azaz $\forall i, j \; x_{ij}, y_{ij}, t_{ij} \geq 0$, illetve $a_{ij} \in \{0, 1\}$, hiszen a_{ij} bináris.

A hiányzó feltételek már trükkösebbek. Vegyük például a maximum mennyiség feltételét. Hogyan vegyük figyelembe, hogy ha csak egy terméket gyártanak egyszerre?

A megoldás hasonló az 5.1. példa megoldáshoz, vagyis itt is azt kell kitalálni, hogy a megadott értékből kiszámolható egy termék előállítási ideje. Ugyanis ha vesszük az egy nap alatt maximálisan előállítható mennyiség reciprokát, akkor megkapjuk, hogy egy darab a nap hányadrésze alatt készül el. Ekkor az x_{ij}/db_j az i. termék előállításával töltött napok számát adja meg, vagyis a feltételünk

$$\sum_j \frac{x_{ij}}{db_j} \leq n_i \quad \forall i.$$

Nézzük a minimális mennyiség feltételét. Itt azt gondolhatnánk, hogy az $x_{ij} \geq mm_j$ minden hónapra és minden termékre jó feltételt ad. Ez viszont nem igaz, hiszen ne felejtsük el, hogy ha nem aktiváljuk az adott terméket, akkor nem kell belőle gyártanunk. Vagyis a feltételbe bele kell tennünk, hogy csak akkor kell ez a feltétel, ha az adott termék le is lesz gyártva. Mivel az a_{ij} változónk pont ezt mondja meg, nincs is nehéz dolgunk. Az

$$x_{ij} \geq mm_j a_{ij} \quad \forall i, j$$

pontosan ezt adja meg, hiszen ha $a_{ij} = 1$ akkor a feltétel éles, ha $a_{ij} = 0$ akkor az $x_{ij} \geq 0$ triviális feltétellé válik. Ráadásul ez is lineáris feltétel, ami ugye a megoldhatóság miatt fontos.

Aki óvatlan, azt gondolhatja, hogy ezzel meg is volnánk, hiszen minden információ szerepel valamely feltételben. Igen ám, de vegyük szemügyre a feltételeinket (rosszabb esetben már az optimalizálás eredményét), és vegyük észre, hogy valami nem stimmel. Ugyanis jelenleg x_{ij} lehet nulla, y_{ij} lehet d_{ij}, és t_{ij} is bármi. Pedig ezek között a változók között fontos összefüggés van: annyit adunk el, amennyit megtermeltünk illetve a raktárba volt, kivéve amennyit most a raktárban hagyunk. Képlettel,

$$y_{ij} + t_{ij} = t_{i-1,j} + x_{ij} \quad \forall i, j,$$

ahol $i - 1$ természetesen az előző hónap, ami egyben azt is jelenti, hogy mindenképpen számokkal adjuk meg a hónapokat, illetve gondoskodnunk kell a 0. hónapban a tárolt mennyiség 0-ra állításáról is, azaz $t_{0j} = 0 \; \forall j$. Ezt egyensúlyi feltételnek is hívhatjuk.

Most újra végignézve a feltételeket, az tűnhet fel, hogy ugyan az a_{ij} változók szerepelnek egy feltételben, de az nem biztosítja, hogy ha $x_{ij} > 0$ akkor $a_{ij} = 1$ legyen. Márpedig nekünk erre szükségünk van. Ha saját kútfőből akarjuk kitalálni a megfelelő feltételt, akkor induljunk ki például az $a_{ij} \geq x_{ij}$ feltételből, ami természetesen nem jó ebben a formában, de jóvá tehető. Láthatjuk, hogy ha mondjuk $a_{ij} \geq \frac{x_{ij}}{M}$ lenne, ahol M egy nagyon nagy szám, akkor már triviálisan jó is. Viszont M helyett mondhatunk jobbat, hiszen elég ha x_{ij}-nél adunk nagyobbat, és nem túl sokkal, különben nehezítjük a feladatunk (minél élesebb egy feltétel, annál jobb). Vagyis mi x_{ij} maximális értéke? Lehetne egyrészt az adott termék összkereslete (a havi nem elég, mert gyárthatunk előre is), de még jobb a maximális napi darabszám szorozva a hónap napjainak számával,

azaz

$$a_{ij} \geq \frac{x_{ij}}{db_j n_i} \quad \forall i,j.$$

Ezzel kész is volnánk. Most foglaljuk össze a matematikai modellt.

$$\begin{aligned}
y_{ij} &\leq d_{ij} && \forall i,j \\
\sum_j t_{ij} &\leq 800 && \forall i \\
\sum_j \frac{x_{ij}}{db_j} &\leq n_i && \forall i \\
mm_j a_{ij} &\leq x_{ij} && \forall i,j \\
y_{ij} + t_{ij} &= t_{i-1,j} + x_{ij} && \forall i,j \\
\frac{x_{ij}}{db_j n_i} &\leq a_{ij} && \forall i,j \\
a_{ij} \in \{0,1\},\ x_{ij}, y_{ij}, t_{ij} &\geq 0,\ t_{0j} = 0 && \forall i,j \\
\hline
\max \sum_i \left(\sum_j ar_j y_{ij} - \sum_j gyk_j x_{ij} - \sum_j tk_j t_{ij} - \sum_j ak_j a_{ij} \right)
\end{aligned} \tag{5.1}$$

Az AMPL felírást kezdhetjük az adatok definiálásával, mint ahogy a jelölést bevezettük a modell előtt. Az 5.4. kódlistában láthatjuk, hogy a hónapokat speciálisan definiáltuk. Egyrészt nem csak megadtuk, hogy ez egy halmaz, hanem mindjárt azt is közöltük, hogy 1-től a hónapok számáig minden egész szám az eleme ennek a halmaznak. Vagyis a `set HONAPOK:=1..Honapok;` parancs ezt jelenti. Itt állhat `a..b`, vagy `1991..2013` is, a két pont a matematikai „ ,..., ”-őt helyettesíti. Még ezt lehet fokozni azzal, hogy ne egy legyen a különbség két egymást követő tag között, hanem mondjuk 10, ha a végére a `by 10` kiegészítést tesszük (például `set N3:=0..333 by 3`). A következő megjegyzés az indexhalmazoknál a `HONAPOK0` definícióját illeti (ez majd az egyensúlyi feltételhez kell). Ezt megadhattuk volna a `set HONAPOK0:=0..Honapok;` paranccsal is, de így megtanuljuk, hogy van `union`, `inter` és `diff` parancs is, amivel két halmaz unióját, metszetét, illetve különbségét definiálhatjuk, illetve, hogy egy halmaz megadható kapcsoszárójelek között az elemek megadásával.

A paraméterek megadásánál annyit érdemes megjegyeznünk, hogy itt is tehetünk feltételeket a paraméterekre, mint például a nemnegativitási feltétel a példában, de persze ez inkább csak önellenőrzés miatt fontos, hogy a fordító azonnal kiabáljon, ha nem jó értéket aduk az adatfájlban egy paraméternek. Láthatjuk továbbá, hogy mátrix adatok esetén egyszerűen csak megadjuk a két indexhalmaxt, ami felett definiált, és többdimenziós tömbök esetén hasonlóan járunk el csak több halmazzal.

5.4. Kódlista: prod.mod

```
set TERMEK;
param Honapok;
set HONAPOK := 1..Honapok;
set HONAPOK0 := HONAPOK union {0};

param napok{HONAPOK} >= 0;
param kereslet { TERMEK, HONAPOK } >= 0;
param ar { TERMEK } >= 0;
param koltseg { TERMEK } >= 0;
param maxegynap { TERMEK } >= 0;
param aktivalas { TERMEK } >= 0;
param minimum { TERMEK } >= 0;
param tarolas { TERMEK } >= 0;
param kapacitas >= 0;

var gyartott { TERMEK, HONAPOK } >= 0, integer;
var eladott { TERMEK, HONAPOK } >= 0, integer;
var tarolt { TERMEK, HONAPOK0 } >= 0, integer;
var aktivalt { TERMEK, HONAPOK } >= 0, binary;

maximize profit:
sum {i in TERMEK}
(ar[i] * sum {j in HONAPOK} eladott[i,j] -
 koltseg[i] * sum {j in HONAPOK} gyartott[i,j] -
 tarolas[i] * sum {j in HONAPOK} tarolt[i,j] -
 aktivalas[i] * sum {j in HONAPOK} aktivalt[i,j]) ;

subject to maxeladas {i in TERMEK, j in HONAPOK}:
eladott[i,j] <= kereslet[i,j];

subject to maxgyartas {j in HONAPOK}:
sum {i in TERMEK} (gyartott[i,j] / maxegynap[i]) <= napok[j];

subject to egyensuly {i in TERMEK, j in HONAPOK}:
tarolt[i,j-1] + gyartott[i,j] = tarolt[i,j] + eladott[i,j];

subject to tarolokapacitas {j in HONAPOK}:
sum {i in TERMEK} tarolt[i,j] <= kapacitas;

subject to aktive {i in TERMEK, j in HONAPOK}:
gyartott[i,j] <= napok[j]*maxegynap[i]*aktivalt[i,j];

subject to minimalistermeles {i in TERMEK, j in HONAPOK}:
gyartott[i,j] >= minimum[i]*aktivalt[i,j];
```

A változók definiálásánál nincs nagy újdonság, ami fontos, hogy a `tarolt` változók a kiterjesztett `HONAPOK0` indexhalmazon fut, hiszen az egyensúlyi feltételnél megjelennek a t_{0j} változók (de a feltételben csak a `HONAPOK` halmazon fut az i, hiszen így veszi fel az $i-1$ a 0 értéket).

Az így megadott modell fájlból már csak a kezdő raktárkészlet értékek megadása hiányzik, de erre mutatunk alternatívát az adatfájlban. Az 5.5. kódlista végén mint kezdőértékeket adjuk meg a nullákat a `let` paranccsal, majd a `fix` paranccsal fixáljuk le ezeket. Itt láthatjuk még a kereslet mátrix szintaktikai megadását, majd alatta a termékekre definiált vektorokét is. Figyeljük meg, hogy a két adatblokk között csak annyi a különbség hogy a keresletnél a „:" előtt megadtuk a mátrix nevét, míg a vektoroknál nincs ilyen, így a kettőspont után oszlopindexek helyett szerepelnek a vektorok nevei.

5.5. Kódlista: prod.dat

```
set TERMEK := A1 A2 A3 ;

param Honapok := 4 ;

param napok :=
1 23
2 20
3 23
4 22 ;

param kereslet: 1 2 3 4 :=
A1 5300 1200 7400 5300
A2 4500 5400 6500 7200
A3 4400 6700 12500 13200 ;

param : ar koltseg maxegynap aktivalas minimum tarolas :=
A1 124 73.30 500 150000 20 3.5
A2 109 52.90 450 150000 20 4
A3 115 65.40 550 100000 16 3 ;

param kapacitas := 800 ;

let {i in TERMEK} tarolt[i,0] := 0;
fix {i in TERMEK} tarolt[i,0];
```

A futtató fájlba csak egy újdonságot tettünk. Az `option show_stats 1;` parancs segítségével beállítjuk, hogy a futás során a megoldó írja ki az előfeldolgozó

(presolve) statisztikáját, vagyis azt, hogy hány változót, illetve feltételt tudott törölni az eredeti feladatból.

5.6. Kódlista: prod.run

```
model prod.mod;
data prod.dat;
option solver cplex;
solve;
option display_round 4;
option show_stats 1;
display profit, gyartott, aktivalt, tarolt, eladott;
```

A fenti fájlokra a CPLEX megoldó a következő eredményt adja. A megoldó 31 MIP szimplex lépést tett, és nem volt szükség részproblémák megoldására. Az előfeldolgozó 12 feltételt és 3 változót tudott törölni a modellből. Az optimális profit 1 581 550 $, A3 maximális gyártása mellett, úgy, hogy januárban A1-et, februárban A2-t gyártunk mellette. A megoldásban ellenőrizhető, hogy jó az aktiválási változók értéke, illetve a gyártás-eladás-tárolás is összhangban van, vagyis az AMPL-es megadásunk is jól sikerült.

```
user@pc:/home/user$ ampl prod.run
CPLEX 12.5.1.0: optimal integer solution; objective 1581550
31 MIP simplex iterations
0 branch-and-bound nodes

Presolve eliminates 12 constraints and 3 variables.
Adjusted problem:
48 variables:
        12 binary variables
        36 linear variables
44 constraints, all linear; 117 nonzeros
        12 equality constraints
        32 inequality constraints
1 linear objective; 48 nonzeros.

profit = 1581550

:      gyartott aktivalt tarolt   eladott     :=
0 A1       .         .       0        .
0 A2       .         .       0        .
```

```
0 A3        .          .        0         .
1 A1     6100          1      800      5300
1 A2        0          0        0         0
1 A3     4400          1        0      4400
2 A1        0          0        0       800
2 A2     3518.18       1        0      3518.18
2 A3     6700          1        0      6700
3 A1        0          0        0         0
3 A2        0          0        0         0
3 A3    12650          1      150     12500
4 A1        0          0        0         0
4 A2        0          0        0         0
4 A3    12100          1        0     12250
;
```

6. fejezet

A GAMS leíró nyelv

6.1. Példa. Vegyük a 2.3. példát az egyszerűség kedvéért. A szállítási feladathoz tartozó táblázatokat most írjuk le egyben az 6.1. táblázatban, de adjunk nevet az egyes raktáraknak és diszkontoknak.

6.1. táblázat. Szállítási költségek

	Diszkontok				
Raktárak	Szeged	Szolnok	Debrecen	Gyula	Raktárkészlet
Cegléd	132	-	97	103	135
Makó	85	91	-	-	56
Kecskemét	106	89	100	98	93
Megrendelések	62	83	39	91	

Célunk a megrendelések kielégítése a legkisebb szállítási költséggel.

Megoldás: A példa egy egyszerű szállítási feladat, aminek modelljét az 2.3. példa megoldásában már megadtuk, így kezdhetjük a GAMS-os felírással!

6.1. A szállítási GAMS modell

Mint látni fogjuk a GAMS leíró nyelv sokban hasonlít, de sokban különbözik is az AMPL-től. Mivel már ismerjük a modellek alapvető elemeit, most ezek szerint fogunk sorban haladni. Nézzük először az indexhalmazokat.

Indexhalmazok

A halmazokat itt is a `SET` vagy `SETS` kulcsszóval tudjuk megadni (a kettő között nincs különbség). Viszont GAMS-ban lehetőségünk van a halmazok (illetve más elemek) együttes definiálására is, de ilyenkor nagyon fontos, hogy csak az utolsó után tegyük ki a pontosvesszőt, hiszen ott van a parancs vége. GAMS-ban a kis és nagybetűk között nem teszünk különbséget, mint az AMPL esetén.

6.1. Kódlista: GAMS indexhalmaz definíciók

```
SETS
     RAK   raktárak ahonnan szállítunk
                  / Cegled, Mako, Kecskemet /
     DIS   "diszkontok, ahova szállítunk"
                 / Szeged, Szolnok, Debrecen, Gyula / ;
```

Alapvető különbség, hogy GAMS-ban mindig van lehetőségünk magyarázó szöveget írni az elemekhez, mint például a 6.1. kódlistában láthatjuk, a két halmazunk után közvetlenül megadtuk a halmazok jelentését. Ezeket az eredmény jelentésben viszontlátjuk majd, ami egy összetett modell esetén sokban könnyíti az eredmény értelmezését. Ilyen magyarázat minden elem esetén megadható közvetlen a név után, viszont figyelnünk kell arra, hogy ha vesszőt, zárójelet, vagy más extra karaktert írnánk a magyarázó szövegbe, akkor mindenképpen tegyük idézőjelbe, mint esetünkben a diszkontok definíciójánál van. A magyarázó szövegbe írhatunk ékezeteket, ezeket máshol kerüljük, mert hibaüzenetet adnak.

Vegyük észre, hogy az AMPL-lel ellentétben itt rögtön megadjuk az adatokat is, `/`-ek között felsorolva, vesszővel, vagy újsorral elválasztva. Ez persze az általános megadás, megtehetjük, hogy csak definiáljuk, majd később adjuk meg, de mivel egy fájlban van, itt ez a célszerűbb.

Paraméterek

GAMS-ban a `PARAMETER` vagy `PARAMETERS` kulcsszóval adhatjuk meg a vektorainkat. Itt jegyezzük meg, hogy skalárok esetén itt külön kulcsszó szükséges, a `SCALAR` vagy `SCALARS`. Hasonlóan az AMPL-hez, itt is megadjuk a vektorok neve után az indexhalmazt, aminek minden elemére definiáljuk a vektort, majd „index érték” párokban adunk értéket. Viszont, amíg AMPL-ben fontos volt annak a megadása, hogy melyik index melyik halmaz eleme (például `i in I`), addig GAMS-ban egyszerűen a halmaz nevét használjuk. Ez könnyebbséget, de egyben nehézséget is jelent. Amíg egyszerű esetben nem kell megadni futóindexet, addig ha két különböző futó indexet használnánk ugyanarra a halmazra, akkor trükkös megoldáshoz kell folyamodnunk. Ugyanis, ha mondjuk az $x_i x_j$ szorzatra lenne szükségünk ($i, j \in I$), akkor az

6.2. Kódlista: GAMS paramétervektorok megadása.

```
PARAMETERS
     KESZLET(RAK)  az i-edik raktár készlete
       /   Cegled     135
           Mako        56
           Kecskemet   93   /
     MEGREND(DIS)  a j-edik diszkont megrendelése
       /   Szeged     62
           Szolnok    83
           Debrecen   39
           Gyula      91   / ;
```

`x(I)*x(J)` nem jó, hiszen a `J` halmaz nem definiált, az `x(I)*x(I)` pedig nyilván nem a jó eredményt adja. Ezért találták ki az `ALIAS` parancsot, amivel egy halmazra több néven is hivatkozhatunk. Tehát az `ALIAS(I,J);` parancs után az `x(I)*x(J)` már használható lesz.

Táblázat

Itt a táblázatok megadása is külön történik a `TABLE` kulcsszóval, de a táblázatokat csak egyesével lehet megadni. Az AMPL-lel ellentétben itt nem az a fontos, hogy az adatok szép sorban meg legyenek adva akárhány szóköz vagy tabulátor elválasztásával, hanem hogy az oszlopindex és a hozzá tartozó értéknek legyen közös oszlopmetszete. Így megengedhető, hogy bizonyos adatok ne legyenek kitöltve, mint például a 6.3. kódlistában. A meg nem adott értékek mindegyike 0 lesz.

6.3. Kódlista: GAMS mátrix megadás.

```
TABLE
      C(RAK,DIS)  a szállítás egységköltségei
                    Szeged   Szolnok  Debrecen    Gyula
        Cegled        132                 97       103
        Mako           85       91
        Kecskemet     106       89       100        98  ;
```

Hogy lássuk ebben a megadásban rejlő hibalehetőségeket, készítettünk egy rossz mátrixot is a 6.4. kódlistában. Itt szürkével jelöltük minden oszlopra azt a sávot, ahova a megfelelő adatoknak kerülniük kell. Vegyük észre, hogy a piros háttérben szereplő számoknak nincs megfelelő oszlopuk, így erre hibaüzenetet kaphatunk, illetve a sárgával jelölt érték két oszlopba is benyúlik, így ez is hibás. Előfordulhat, hogy a

6.4. Kódlista: Hibás mátrix megadás.

```
TABLE
      ROSSZ(I,J)  hibás megadások
                        I.          II.     Harmadik
        a           13200         10000000    97
        b             85.0        91
        c                         89          100       98  ;
```

szövegszerkesztőnk jól mutatja a táblázatot, mégis hibákat jelez a fordító. Ilyenkor a tabulátorok használatát kerüljük, mert sajnos azok hossza szerkesztőnként más és más, így érdemesebb inkább csak szóközökkel állítani az oszlopokat.

A fenti példánkban a nem megadott szállítási költségek azt akarják jelezni, hogy ott nem történhet szállítás. A GAMS alapértelmezett 0 értékadása tehát itt nekünk nem jó, így adjuk meg, hogy a 0 helyett legyen 1000 a szállítási költség ezeken az utakon. Ehhez használhatjuk a GAMS `$` feltételét. A `$(FELT)` feltétel sok helyen állhat, és lényegében a parancs csak akkor hajtódik végre ha a feltétel teljesül. Esetünkben tehát használhatjuk a `C(RAK,DIS) $( C(RAK,DIS)=0 ) = 1000;` parancsot. Megjegyezzük viszont, hogy ha a `C(RAK,DIS) = 1000$( C(RAK,DIS)=0 );` parancsot adtuk volna, akkor a költség az eddig definiált helyeken 0-ra változna, mivel az egyenlet jobb oldalán áll a feltétel.

Változók

A változók definiálása hasonló az eddig látottakhoz, és a `VARIABLE` vagy `VARIABLES` kulcsszavakkal történik. A változók típusát megadhatjuk a definíció után, de akár definiálhatjuk is mindjárt a típusának megfelelően. Egy változó lehet `FREE` előjelkötetlen, `POSITIVE` nemnegatív, `NEGATIVE` nempozitív, `BINARY` bináris, vagy `INTEGER` egész (0,1,...,100 ha másképp nem definiáljuk). Az alapértelmezett természetesen az előjelkötetlen. A 6.5. kódlistában megadtuk a feladatunkban szereplő változókat. A `Z`

6.5. Kódlista: GAMS változók megadása.

```
VARIABLES
     X(RAK,DIS)  a szállított mennyiségek
     Z           a teljes szállítási költség ;
POSITIVE VARIABLE X ;
```

változónk tulajdonképpen a célfüggvény lesz, ez szintén a GAMS sajátossága, hogy a célfüggvényt egy változóval egyenlővé téve a változót fogjuk optimalizálni.

Korlátozó feltételek

A célfüggvényt és a korlátozó feltételeket az `EQUATION` és `EQUATIONS` kulcsszavakkal adhatjuk meg. Ahogy azt a 6.6. kódlistában is láthatjuk, először definiáljuk az egyes feltételeket és a célfüggvényt, majd külön-külön megadjuk a hozzájuk tartozó képleteket. A képletek megadása szintaktikailag úgy történik, hogy a feltétel (vagy célfüggvény) neve után pontosan két ponttal elválasztva (tekinthetjük ezt egy fektetett kettőspontnak) adjuk meg a megfelelő formulát. Itt az `=E=` jelentése egyenlő, az `=L=`

6.6. Kódlista: A célfüggvény és a feltételek definiálása GAMS-ban.

```
EQUATIONS
     COST         célfüggvény
     SUPPLY(RAK)    a raktárkészletek korlátozó feltétele
     DEMAND(DIS)    a diszkontok megrendelésének feltétele;

COST ..    Z  =E=  SUM( (RAK,DIS) , C(RAK,DIS)*X(RAK,DIS) );
SUPPLY(RAK) ..   SUM(DIS, X(RAK,DIS))  =L=  KESZLET(RAK);
DEMAND(DIS) ..   SUM(RAK, X(RAK,DIS))  =E=  MEGREND(DIS);
```

jelentése kisebb vagy egyenlő, míg az `=G=` jelöli a nagyobb vagy egyenlő relációt. A szumma szintaktikája is egy kicsit más itt, a `SUM( MIRE , MIT )` példában a `MIRE` halmaz minden elemére összegezzük a `MIT`. Természetesen ha ez utóbbi függ az adott vagy akár más halmaztól akkor azt zárójelben jelezzük, mit ahogy a 6.6. kódlistában is tettük az egyenlőtlenségeink megadásánál. A célfüggvény megadásánál azt vehetjük észre, hogy egyszerre adhatunk meg egy dupla szummát úgy, hogy a `MIRE` halmazt `(RAK,DIS)` formában adjuk meg.

A modell leírása és a feladat megoldása

A modell megadása a `MODEL TRANSPORT /ALL/;` paranccsal történhet. Itt a modellnek adunk egy nevet (most `TRANSPORT`), és megadjuk, hogy mely korlátozó feltételeket kell figyelembe venni. Ez esetben az összes feltételt bevesszük, erre utal az `ALL`, amely egyenértékű lenne a `MODEL TRANSPORT /COST, SUPPLY, DEMAND/` megadással.

A feladat megoldását a `SOLVE TRANSPORT USING LP MINIMIZING Z;` paranccsal kérhetjük. Ebben a sorban adjuk meg, hogy melyik modellt oldja meg, itt a `TRANSPORT` modellt, hogy milyen megoldó programmal, itt LP feladatot adtunk meg, így `LP` kell, hogy minimalizáljon vagy maximalizáljon, `MINIMIZING` vagy `MAXIMIZING`, illetve hogy melyik változó szerint, azaz mire adtuk meg a célfüggvényt (itt `Z`).

Nézzük meg most egyben a GAMS modellünket a 6.7. kódlistában.

6.7. Kódlista: transport.gms

```
SETS
     RAK   raktárak ahonnan szállítunk
                   / Cegled, Mako, Kecskemet /
     DIS   "diszkontok, ahova szállítunk"
                  / Szeged, Szolnok, Debrecen, Gyula / ;

PARAMETERS
     KESZLET(RAK)  az i-edik raktár készlete
       /   Cegled     135
           Mako        56
           Kecskemet   93   /
     MEGREND(DIS)  a j-edik diszkont megrendelése
       /   Szeged      62
           Szolnok     83
           Debrecen    39
           Gyula       91   / ;

TABLE
      C(RAK,DIS)  a szállítás egységköltségei
                    Szeged   Szolnok  Debrecen    Gyula
        Cegled        132                97        103
        Mako           85       91
        Kecskemet     106       89       100         98  ;

C(RAK,DIS) $( C(RAK,DIS)=0 ) = 1000;

VARIABLES
     X(RAK,DIS)  a szállított mennyiségek
     Z       a teljes szállítási költség ;
POSITIVE VARIABLE X ;

EQUATIONS
     COST        célfüggvény
     SUPPLY(RAK)   a raktárkészletek korlátozó feltétele
     DEMAND(DIS)   a diszkontok megrendelésének feltétele;

COST ..   Z  =E=  SUM( (RAK,DIS) , C(RAK,DIS)*X(RAK,DIS) );
SUPPLY(RAK) ..   SUM(DIS, X(RAK,DIS))  =L=  KESZLET(RAK);
DEMAND(DIS) ..   SUM(RAK, X(RAK,DIS))  =E=  MEGREND(DIS);

MODEL TRANSPORT /ALL/;

SOLVE TRANSPORT USING LP MINIMIZING Z;
```

6.2. A GAMS megoldása

Ha telepítve van a GAMS a számítógépünkön (ellenkező esetben lásd a A.3. függeléket), akkor a `gams transport.gms` paranccsal oldhatjuk meg a feladatunkat. A parancsot kiadva a következő választ kapjuk.

```
--- Job transport.gms Start 08/14/13 15:55:22 LEX-LEI 23.6.3 x86_64/Linux
GAMS Rev 236  Copyright (C) 1987-2011 GAMS Development. All rights reserved
Licensee: GAMS Development Corporation, Washington, DC   G871201/0000CA-ANY
          Free Demo,  202-342-0180,  sales@gams.com,  www.gams.com   DC0000
--- Starting compilation
--- transport.gms(43) 3 Mb
--- Starting execution: elapsed 0:00:00.024
--- transport.gms(25) 4 Mb
--- Generating LP model TRANSPORT
--- transport.gms(43) 4 Mb
---   8 rows  13 columns  37 non-zeroes
--- transport.gms(43) 4 Mb
--- Executing XPRESS: elapsed 0:00:00.119
--- transport.gms(43) 4 Mb

FICO-Xpress      Dec 13, 2010 23.6.3 LEX 22848.22869 LEI x86_64/Linux

Xpress Optimizer 21.01
Xpress Optimizer 64-bit v21.01.00 (Hyper capacity)
(c) Copyright Fair Isaac Corporation 2010
Licensed for use by: GAMS Development Corp. for GAMS


Reading Problem GAMS Model
Problem Statistics
           7 (      0 spare) rows
          12 (      0 spare) structural columns
          24 (      0 spare) non-zero elements
Global Statistics
           0 entities        0 sets        0 set members
Minimizing LP GAMS Model
Original problem has:
         7 rows           12 cols           24 elements
Presolved problem has:
         7 rows           12 cols           24 elements

   Its         Obj Value      S   Ninf  Nneg        Sum Inf  Time
     0           .000000      D      4     0     275.000000     0
     6       25919.00000      D      0     0        .000000     0
Uncrunching matrix
     6       25919.00000      D      0     0        .000000     0
Optimal solution found
```

```
optimal LP solution found: objective value 25919
--- Restarting execution
--- transport.gms(43) 2 Mb
--- Reading solution for model TRANSPORT
--- transport.gms(43) 2 Mb
*** Status: Normal completion
--- Job transport.gms Stop 08/14/13 15:55:22 elapsed 0:00:00.299
```

A kiírás elején láthatjuk, hogy először lefordítja a GAMS a kódot, generál egy LP feladatot, majd megoldja a 8 feltétellel és 13 változóval rendelkező feladatot, amihez 37 nemnulla együttható tartozik. A 13. változónk itt a célfüggvény változója, és 24 egyes együttható van a feltételek jobboldalán, míg 13 a célfüggvény egyenletében. A megoldó itt a FICO Xpress optimalizálója, ami a további információt adja. A statisztikában láthatjuk, hogy az átadott modell már csak 7 sorral, 12 változóval és 24 nemnulla együtthatóval bír, ez csak annak köszönhető, hogy a célfüggvény sora kikerült a feltételek közül. Az előfeldolgozó (presolve) nem tudta ezt csökkenteni, így a megoldandó probléma is pont ugyan ekkora. Az optimális megoldáshoz tartozó célfüggvényérték 25919, de vigyázzunk, ha a feladatot sikerült az előfeldolgozónak egyszerűsítenie, akkor az itt adott érték az egyszerűsített feladathoz tartozik, ami eltérhet az eredetitől. Így az eredményhez érdemes inkább az eredményfájlt megtekinteni. A GAMS mindig a modellfájl kiterjesztését cseréli le `.lst`-re az eredeti fájl nevében, vagyis most a `transport.lst` fájlt kell megtekintenünk. Mivel ez egyben túl hosszú, így csak egyes részeit emeljük ki.

A fájl elején a GAMS modellünket látjuk viszont, ha a fordító hibát talált akkor az ugyanígy kiírt modellben a hibás sorok alatt számozott kódokkal vannak jelölve a különféle hibák, a modell alatt pedig az egyes hibakódokhoz tartozó üzeneteket olvashatjuk. A GAMS sikeres fordítás esetén kilistázza a feltételeket. Ez esetünkben így néz ki.

```
---- COST  =E=  célfüggvény

COST..  - 132*X(Cegled,Szeged) - 1000*X(Cegled,Szolnok)

      - 97*X(Cegled,Debrecen) - 103*X(Cegled,Gyula)

      - 85*X(Mako,Szeged) - 91*X(Mako,Szolnok)

      - 1000*X(Mako,Debrecen) - 1000*X(Mako,Gyula)

      - 106*X(Kecskemet,Szeged) - 89*X(Kecskemet,Szolnok)

      - 100*X(Kecskemet,Debrecen) - 98*X(Kecskemet,Gyula)

      + Z =E= 0 ; (LHS = 0)
```

```
---- SUPPLY  =L=  a raktárkészletek korlátozó feltétele

SUPPLY(Cegled)..  X(Cegled,Szeged) + X(Cegled,Szolnok) + X(Cegled,Debrecen)
       + X(Cegled,Gyula) =L= 135 ; (LHS = 0)

SUPPLY(Mako)..  X(Mako,Szeged) + X(Mako,Szolnok) + X(Mako,Debrecen)
       + X(Mako,Gyula) =L= 56 ; (LHS = 0)

SUPPLY(Kecskemet)..  X(Kecskemet,Szeged) + X(Kecskemet,Szolnok)
       + X(Kecskemet,Debrecen) + X(Kecskemet,Gyula) =L= 93 ;
      (LHS = 0)


---- DEMAND  =E=  a diszkontok megrendelésének feltétele

DEMAND(Szeged)..  X(Cegled,Szeged) + X(Mako,Szeged)
      + X(Kecskemet,Szeged) =E= 62 ; (LHS = 0, INFES = 62 ****)

DEMAND(Szolnok)..  X(Cegled,Szolnok) + X(Mako,Szolnok)
      + X(Kecskemet,Szolnok) =E= 83 ; (LHS = 0, INFES = 83 ****)

DEMAND(Debrecen)..  X(Cegled,Debrecen) + X(Mako,Debrecen)
      + X(Kecskemet,Debrecen) =E= 39 ; (LHS = 0, INFES = 39 ****)

REMAINING ENTRY SKIPPED
```

Láthatjuk, hogy az eredeti feltételeinket írja ki az együtthatókkal, illetve zárójelben megadja, hogy a baloldal a 0 kezdővektorral milyen értéket vesz fel, `LHS =`, és ha így nem teljesül a feltétel, akkor mekkora az infízibilitás, `INFES =`, ez a megrendelés feltételekben jelentkezik esetünkben.

Ezután következik az oszlopok felsorolása, vagyis az egyes változókról kapunk információt, hogy melyik feltételekben szerepel és milyen együtthatókkal. A feladatunkra ez a következő:

```
---- X  a szállított mennyiségek

X(Cegled,Szeged)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
```

```
     -132       COST
        1       SUPPLY(Cegled)
        1       DEMAND(Szeged)

X(Cegled,Szolnok)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
    -1000       COST
        1       SUPPLY(Cegled)
        1       DEMAND(Szolnok)

X(Cegled,Debrecen)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
      -97       COST
        1       SUPPLY(Cegled)
        1       DEMAND(Debrecen)

REMAINING 9 ENTRIES SKIPPED

---- Z  a teljes szállítási költség

Z
                (.LO, .L, .UP, .M = -INF, 0, +INF, 0)
        1       COST
```

A célfüggvényegyütthatók negatív előjele a minimalizálás maximalizálásra való átírása miatt van.

Ezután következik a megoldás összegzése, ahol először a modell tulajdonságait, majd a megoldó nevét verziószámát olvashatjuk. Utána megkapjuk a megoldó üzenetét, miszerint a feladatunkra optimális megoldást talált 25919 célfüggvényértékkel (optimal LP solution found: objective value 25919). Ezek után következik a megoldás részletesebb megadása. Itt először a célfüggvény és a feltételek néhány érzékenységvizsgálati értékét olvashatjuk. A négy oszlop sorban a jobboldal alsó korlátját, az aktuális értékét, a felső korlátját, és az árnyérárát adja meg az egyes feltételekre. Itt az alsó és felsőkorlát nem a megengedhető növekedést vagy csökkenést jelöli, csak az eredeti korlátokat. A `.` a nullát jelenti minden esetben, míg az `EPS` a kicsi de nem nulla értéket jelöli. Ezt használhatjuk arra is, hogy tudjuk, a nullánál nembázis változó, míg `EPS`-nél bázisváltozó van.

A változók táblázatában hasonlóan a négy oszlopban kapjuk meg egy változó alsó korlátját, aktuális értékét, felső korlátját, és redukált költségét. Ahogy a feltételeknél is, itt sem adja meg a célfüggvényegyütthatóra vonatkozó megengedhető növekedést és csökkenést a táblázat, csak a redukált költséget a nembázis változókra.

```
               S O L V E      S U M M A R Y

     MODEL   TRANSPORT           OBJECTIVE  Z
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  XPRESS              FROM LINE  43

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      1 Optimal
**** OBJECTIVE VALUE            25919.0000

 RESOURCE USAGE, LIMIT          0.038      1000.000
 ITERATION COUNT, LIMIT         6    2000000000

FICO-Xpress      Dec 13, 2010 23.6.3 LEX 22848.22869 LEI x86_64/Linux

Xpress Optimizer 21.01
Xpress Optimizer 64-bit v21.01.00 (Hyper capacity)
optimal LP solution found: objective value 25919

                     LOWER         LEVEL         UPPER       MARGINAL

---- EQU COST          .             .             .           1.0000

  COST  célfüggvény

---- EQU SUPPLY  a raktárkészletek korlátozó feltétele

                 LOWER          LEVEL          UPPER         MARGINAL

Cegled           -INF          126.0000       135.0000          .
Mako             -INF           56.0000        56.0000       -26.0000
Kecskemet        -INF           93.0000        93.0000        -5.0000

---- EQU DEMAND  a diszkontok megrendelésének feltétele

                LOWER          LEVEL          UPPER         MARGINAL

Szeged          62.0000        62.0000        62.0000       111.0000
Szolnok         83.0000        83.0000        83.0000        94.0000
Debrecen        39.0000        39.0000        39.0000        97.0000
Gyula           91.0000        91.0000        91.0000       103.0000

---- VAR X  a szállított mennyiségek

                      LOWER       LEVEL        UPPER       MARGINAL

Cegled   .Szeged        .           .          +INF         21.0000
Cegled   .Szolnok       .           .          +INF        906.0000
Cegled   .Debrecen      .         39.0000      +INF           .
```

```
Cegled    .Gyula          .          87.0000      +INF            .
Mako      .Szeged         .          56.0000      +INF            .
Mako      .Szolnok        .            .          +INF          23.0000
Mako      .Debrecen       .            .          +INF         929.0000
Mako      .Gyula          .            .          +INF         923.0000
Kecskemet.Szeged          .           6.0000      +INF            .
Kecskemet.Szolnok         .          83.0000      +INF            .
Kecskemet.Debrecen        .            .          +INF           8.0000
Kecskemet.Gyula           .           4.0000      +INF            .

                          LOWER       LEVEL        UPPER       MARGINAL

---- VAR Z               -INF     25919.0000      +INF            .

  Z  a teljes szállítási költség


**** REPORT SUMMARY :        0     NONOPT
                             0 INFEASIBLE
                             0  UNBOUNDED
```

A megoldás után láthatjuk – ami igazán nem optimális esetben lesz érdekes –, hogy nulla a nemoptimális, nemmegengedett, nemkorlátos válasz, vagyis optimális a megoldásunk. Ezek után már csak néhány technikai adat következik, amit itt fel sem soroltunk.

7. fejezet

Megoldók

Az előző fejezetekben láthattuk, hogyan tudjuk egy probléma matematikai modelljét felírni, és a kapott megoldást értelmezni, de nem sok szó esett a megoldás menetéről. Mindegy, hogy az Excel megoldójáról vagy az AMPL-ben, GAMS-ban meghívott megoldóról beszélünk, a problémák megoldásához elengedhetetlen egy jó solver. Persze kis problémákat akár kézzel is megoldhatunk, de minél nagyobb, komplexebb a feladat, annál fontosabb, hogy jó megoldót használjunk. Az Excel beépített megoldója például inkább kisebb, egyszerűbb problémák megoldására használható, míg a Cplex vagy Xpress megoldója képes igazán hatalmas problémák megoldására is.

7.1. Excel Solver

Az alábbiak az Excel súgójában található információk a használható eljárásokról.

A Microsoft Excel Solver eszköz a "Generalized Reduced Gradient" (GRG2) nem lineáris optimalizálási eljárást használja, amelyet Leon Lasdon (University of Texas at Austin) és Allan Waren (Cleveland State University) fejlesztett ki.

A lineáris és az egész értékű problémákra a változókat korlátozó szimplex módszert, valamint a "branch-and-bound" (elágazás és korlátozás) módszert használja; ez utóbbit John Watson és Dan Fylstra (Frontline Systems, Inc.) valósította meg. További részleteket ad a Solver belső megoldási eljárásáról a www.solver.com oldalon találhatunk.

Természetesen nem ez az egyetlen Excellel használható megoldó, viszont ez beépített bárki számára egyszerűen elérhető. Az egyik Excelhez kapcsolható megoldó például a LINDO megoldója, és persze a beépített megoldó fejlesztője is árul komolyabb szoftvert, amit Premium Solver néven kínál.

7.2. A CPLEX megoldó

A CPLEX történetét és lehetőségeit jól összefoglalja a Wikipédia cikke. Az IBM ILOG CPLEX Optimization Studio (gyakran nevezik egyszerűen csak CPLEX-nek) egy optimalizáló szoftver csomag, ami 2004-ben elsőként szerezte meg az INFORMS Impact díját. A CPLEX megoldót a a C programozási nyelven íródott szimplex módszerről nevezték el, de ma már támogat a más típusú matematikai optimalizálási interfészeket is. Eredetileg Robert E. Bixby fejlesztette, 1988-tól árulta a CPLEX Optimization Inc., amelyet megszerzett az ILOG 1997-ben; majd az ILOG-ot később megvásárolta az IBM 2009-ben, ahol továbbra is aktívan fejlesztik.

Az IBM ILOG CPLEX optimalizáló nagy egészértékű programozási feladatok, illetve nagyon nagy lineáris programozási feladatok megoldására is képes a primál vagy a duál szimplex módszer változataival, illetve belső pontos módszerekkel, amivel konvex és nem konvex kvadratikus programozási feladatok, és konvex kvadratikus korláttal rendelkező problémák is megoldhatók.

A CPLEX Optimizer modellező rétege a Concert, amely érintkezik a C++, a C# és a Java nyelvekkel. Van egy Python felülete is, továbbá csatlakozni tud a Microsoft Excel és a MATLAB programokhoz. A CPLEX Optimizer elérhető független modellező rendszerek számára, mint az AIMMS, az AMPL, a GAMS, az MPL, az OpenOpt, az OptimJ és a TOMLAB.

7.3. XPRESS-MP megoldó

Az Xpress Optimization Suite kifinomult, robusztus, többszálú algoritmusok használatával próbálja gyorsan és pontosan megoldani az iparágak legnehezebb problémáit. Bevált optimalizálási technológiát használnak különféle kereskedelmi létesítményekben az egész világon, hogy gyors és megbízható megoldásokat adjanak több millió változó és korlát mellett. Az osztott memóriát és determinisztikus párhuzamosságot kihasználva annyi CPU magot képes kezelni, amennyi elérhető a gyors futás érdekében. Az Xpress-Optimizer ultra hatékony ritka mátrix kezelést és menetközbeni adat tömörítést használ, hogy az ipar által kínált legnagyobb problémákat megoldja. Az Xpress-Optimizer képes megoldani numerikusan nehéz vagy instabil problémákat is, ami az egyik oka annak, hogy egyértelmű piacvezető a feldolgozóiparban.

Az Xpress segítségével a következő problémákat lehet megoldani:

LP - Lineáris problémák

MIP - Vegyes egészértékű problémák

QP - Kvadratikus problémák

MIQP - Vegyes egészértékű kvadratikus problémák

QCQP - Kvadratikus függvényekkel korlátozott kvadratikus problémák

MIQCQP - Kvadratikus függvényekkel korlátozott vegyes egészértékű kvadratikus problémák

NLP - konvex nemlineáris problémák

Amennyiben a probléma nemlineáris és nem konvex, akkor az Xpress-SLP megoldó – amely egymást követő lineáris közelítési technikákat használva – képes megoldani nem-lineáris és vegyes egészértékű nemlineáris problémákat akár több ezer változó mellett.

Az Xpress-Optimizer használható parancssori eszközként egy egyszerű, de hatékony interaktív felhasználói felülettel és meghívható könyvtárként C, C++ , Java, Fortran , VB6 és .NET programozási felületek használatával. Teljes mértékben kompatibilis a szabványos LP és az MPS fájlformátumokkal és átfogó támogatást nyújt naplózáshoz (logoláshoz), bináris mentés, bázis fájlok illetve szöveges/bináris megoldás fájlok használatához.

A hívható könyvtár egyaránt biztosít egy alacsony szintű mátrix orientált API-t és egy objektum orientált modell-építési kezelőfelületet (BCL). Az Xpress szerves része, az Xpress-Mosel, a korszerű modell fejlesztői környezet biztosítja az optimalizáló motor nyers erejének és a teljesítmény beállításának lehetőségét.

7.4. További megoldókról

A NEOS szerver lehetővé teszi, hogy számos megoldót kipróbáljunk, természetesen a feladat típusa szerint. A megfelelő megoldót kiválaszthatjuk a oldalon, amihez segítséget nyújt az úgynevezett Optimization Tree. A honlapon keresztül feltölthetjük a modellünket, ami a NEOS szerveren lesz megoldva a kiválasztott megoldó segítségével. A megoldást a frissülő honlapon, és a megadott e-mail címen is megkapjuk.

8. fejezet

Feladatok

1. Egy gyár négyféle terméket (A, B, C, D) termel három erőforrás (I., II. és III.) segítségével. A fajlagos felhasználásokat, az egyes termékek árát és az egyes erőforrások kapacitását a következő táblázat mutatja:

Erőforrások	Termékek				Erőforrások
	A	B	C	D	kapacitása
I.	1	0	2	1	280
II.	2	1	0	0	140
III.	0	1	1	1	120
Ár	4	5	6	8	

   Mennyit termeljen az egyes termékekből a gyár, ha maximális árbevételt akar elérni az alábbi feltételek teljesülése esetén?

   a) Az erőforrások kapacitása nem léphető túl.
   b) Az A és B termékekből összesen legalább annyit kell termelni, mint a C-ből
   c) A B termékből legfeljebb 5 egységgel termelhető több, mint a D-ből.

2. A Nevenincs ország lényegében négy fő terméket exportál: acélt, motort, elektromos alkatrészeket és műanyagot. A közgazdasági miniszter maximalizálni akarja az exportot és minimalizálni az importot. A világpiaci egységára az acélnak, motornak, elektromos alkatrésznek és műanyagnak a helyi pénznemben (Píz) rendre 500, 1500, 300, és 1200. Egységnyi acél előállításához 0.02 motor, 0.01 egység műanyag, és 250 Píz értékű egyéb importált alapanyag és 6 munkás-hónapnyi munka kell. Egy motor előállításához 0.8 egység acél, 0.15 elektromos alkatrész,

0.11 egység műanyag, 300 Píz értékű importált valami és 1 munkás-év munka kell. Egy elektromos alkatrész előállításához szükséges: 0.01 egység acél, 0.01 motor, 0.05 egység műanyag, 50 Píznyi import anyag és 6 munkás-hónap. Egységnyi műanyaghoz kell 0.03 motor, 0.05 elektromos alkatrész, 0.2 acél, 300 Píznyi import alapanyag és 2 munkás-év. Maximum 650 000 motort, és 60 000 egység műanyagot tudnak gyártani. Az alkalmazott munkások száma maximum 830 000 évente. Acél, motor, elektromos alkatrész és műanyag nem lehet az import része. Adjuk meg a feladat matematikai modelljét ha a profitot maximalizáljuk.

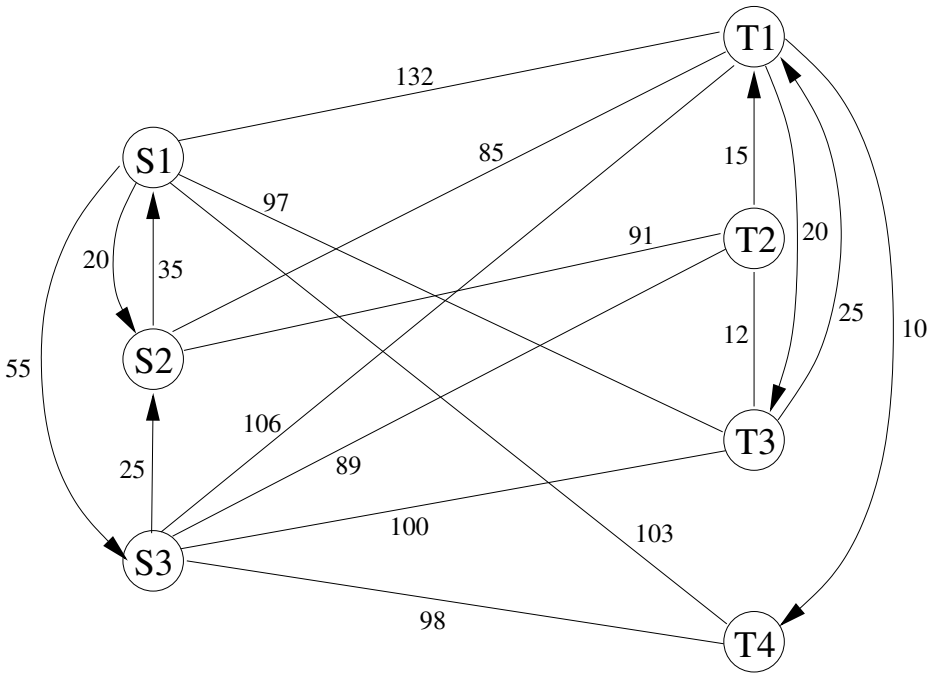
3. Mondjunk olyan típusú nemlineáris függvényt, amelyre az Excel biztos megtalálja a globális minimumot! Miért igaz ez?

4. Három raktár (S1,S2,S3) szolgálja ki négy diszkont (T1,T2,T3,T4) igényeit. A raktárak kapacitása és a diszkontok megrendelése a következő:

S1	S2	S3
135	56	93

T1	T2	T3	T4
62	83	39	91

Az egységnyi szállítás árait az alábbi gráf tartalmazza.



Hogyan telepítsük az összes megrendelést minimális költséggel és a raktárak kapacitásának betartásával?

5. A Hajós utcában mindkét oldalon lehet parkolni. Dr. Újvári partit rendez kb. 30 embernek akik 15 kocsival jönnek. A kocsik hossza méterben a következő:

4	4.5	5	4.1	2.4	5.2	3.7	3.5	3.2	4.5	2.3	3.3	3.8	4.6	3

   Hogy minél kevésbé zavarják a szomszédokat, Dr. Újvári szeretné úgy elrendezni a parkolást hogy minél kisebb szakaszon foglalják el az utcát. Feltesszük, hogy az utca üres és az autók hosszába a minimális parkolási távolság is bele van számolva.

6. Egy számítógépgyártó vállalat négyféle kalkulátort gyárt: a C1 és C2 típusokat házi használatra és a C3 és C4 típusokat tudományos célokra. A számítógépek elkészítéséhez három különböző integrált áramkörre és munkaerőre van szükség. Az A és B áramköröket importálják, a C áramkört a vállalat maga állítja elő. Az alábbi táblázat az egyes gépekhez felhasznált áramkörök számát, a munkaerő-szükségletet (órában) és a nyereséget (ezer Ft-ban) tartalmazza:

	C1	C2	C3	C4
A	5	3	2	0
B	0	0	3	8
C	1	4	6	2
Munkaerő	2	3	4	6
Nyereség	10	30	50	100

   A termeléshez minden periódusban 240 A és 320 B áramkör áll rendelkezésre, a munkaerő korlátja 180 óra. Mi az optimális termelés ha a vállalat célja

   a) a profit maximalizálása,

   b) az otthoni számítógépek gyártásának maximalizálása,

   c) a saját előállítású C áramkör felhasználásának maximalizálása?

7. Írj fel matematikai modellt a 8 királynő problémára, azaz egy sakktáblán úgy helyezzünk el 8 királynőt, hogy egyik se üsse a másikat: se sorban, se oszlopban, se átlósan.

8. Három alkatrészt (A1,A2,A3) három gépen (G1,G2,G3) lehet megmunkálni. Mivel a gépek különböző típusúak és különböző életkorúak, azért az egyes alkatrészek megmunkálásának fajlagos időszükséglete az egyes gépeken különböző.

Az alábbi táblázat mutatja az egyes gépeken egy alkatrész megmunkálásához szükséges időt (órában), az egyes alkatrészek eladási árát, valamint egy gépóra költségét (száz forintban) és az egyes gépek gépóra-kapacitását:

Gépek	Alkatrészek A1	A2	A3	1 gépóra költsége	Gépek kapacitása
G1	0,2	0,1	0,05	30	40
G2	0,6	0,3	0,2	10	60
G3	0,2	0,1	0,3	20	30
Ár	10	16	12		

Feltéve, hogy mindegyik gép bármelyik alkatrész megmunkálására alkalmas, és az A1 alkatrészből legfeljebb annyit szabad megmunkálni, mint a másik kettőből összesen; írja fel annak a termelési programnak megfelelő modellt, amely annak a pénzösszegnek a maximumát eredményezi, amelyet úgy kapunk, hogy az árbevételből levonjuk a gépóraköltséget.

9. Egy cég egy árucikk gyártását kétféleképpen oldja meg: normál és túlórás "műszakban" hogy teljesíteni tudja a jelenlegi és a jövőbeli keresletet. Az adott árucikk előállításához havonta különböző mennyiségű munkaerő áll rendelkezésünkre. Januárban 10, februárban 15, márciusban 14 és áprilisban 16 munkást tudunk erre a munkára állítani. Egy munkás normál műszakban 10 000 egységet állít elő egy hónapban, míg túlórában ennek a felét. A kereslet januárban 80 000, februárban és áprilisban 200 000, míg márciusban 300 000. A normál munkában egy egység előállítása 1 euroba, túlórában 1.5 euroba kerül. 30 centért tárolhatunk egységenként és havonta előre legyártott árut. Hogyan termeljünk, hogy minden kereslet ki legyen elégítve minimális költségen?

10. Milyen nemlineáris függvényre nem találja meg mindig a globális optimumot az Excel Solver? Miért?

11. Időszámításunk előtt 435-ben Spárta elhatározta, hogy attól kezdve tartalékos katonák behívásával erősíti meg a reguláris seregét. Az új harcosok 1, 2 vagy 3 évre hívhatók be. Legyen x_{1T}, x_{2T} és x_{3T} a T-edik évben 1, 2 és 3 éves szolgálatra behívott tartalékos katonák száma. Ezek költsége legyen rendre c_{1T}, c_{2T} és c_{3T}. 5 évre előre minden évre meghatározták azt az RT értéket, ahány főből kell legalább állnia a tartalékos harcosok seregének. Mint spártai hadvezér, lineáris programozási feladat megoldásával határozzon meg egy öt esztendőre szóló besorozási taktikát úgy, hogy minimális költséggel biztosítsa, hogy mind az öt

évben kellő számú tartalékos sereg álljon rendelkezésre. Az egyszerűség kedvéért a T=1 érték tartozzon az időszámítás előtti 435-ös évhez.

Az év kezdetek kori besorozási költségek és a tartalékos állomány létszám igényeinek a táblázata:

	1. év	2. év	3. év	4. év	5. év
Besorozás 1 évre	1000	1200	1500	1800	2000
Besorozás 2 évre	1800	2100	2400	2600	–
Besorozás 3 évre	2000	2400	2800	–	–
Létszám igény	100	120	110	140	130

12. 7 feladatot 3 számítógépen kell megoldani, amiből egy 1.33 GHz, a másik kettő 2.66 GHz. Az elvégzendő elemi műveletek billió instrukcióban megadva:

1.1	2.1	3	1	0.7	5	3

Melyik feladatot melyik gépen végezzük el hogy az utolsó minél hamarabb befejezze?

13. A Ferihegyi repülőtéren a MALÉV légitársaságnak a nap adott időpontjaitól függően változó számú földi kiszolgáló személyzetre van szüksége. A minimális követelményeket az alábbi táblázat tartalmazza:

A nap időintervallumai	Földi személyzet száma
0:00–4:00	7
4:00–8:00	25
8:00–12:00	30
12:00–16:00	5
16:00–20:00	35
20:00–24:00	15

Tegyük fel, hogy a földi személyzet háromfajtaképpen dolgozik. Van a normál műszaki, aki a táblázatban felsorolt hat időintervallum bármelyikének a kezdetekor munkába állítható és ezt követően munkában is marad nyolc egymás utáni órán keresztül. Vannak a félállásúak, akik hasonlóan dolgoznak a normál műszakhoz, de csak 4 órán keresztül, és végül a félnaposok, akik 12 egymást követő órában dolgoznak, de csak kétnaponta. A félállásúak a normál műszakiak 65%-át keresik meg egy hónapban, míg a félnaposok ugyanannyit.

Hányan dolgozzanak milyen műszakban, hogy minimális költséggel a nap minden időszakában rendelkezésre álljon a szükséges számú földi kiszolgáló személyzet?

14. Egy vállalatnak 5 vidéki telephelyet kell létesítenie. Az egyes telephelyek területigénye (ha-ban) az alábbi: 5, 7, 10, 8, 6. A telepítésre 6 telephely jöhet szóba, amelyek területe (ha-ban): 11, 8, 7, 7, 10, 9.

    A telepítéssel kapcsolatos költségeket az alábbi táblázat mutatja:

Üzem	Telephely					
	1.	2.	3.	4.	5.	6.
I.	15	18	30	25	22	28
II.	19	27	25	23	24	24
III.	26	22	20	20	17	12
IV.	20	21	21	20	19	23
V.	22	16	15	18	20	24

    Egy telephelyre legfeljebb egy üzemet lehet telepíteni. Mely telepítési terv mellett lesz a telepítéssel kapcsolatos költségek összege minimális?

    Írd fel a matematikai modellt, és old meg a feladatot AMPL-ben!

15. Egy cég 5 különböző anyagot fest 3 kádban. Minden anyagnak 2 kádban kell fürödnie, de mindegy, hogy melyik kettőben és hogy milyen sorrendben. Az anyagok a különböző fürdőkben különböző ideig áznak, amit a következő táblázat ad meg órákban.

	1.F	2.F	3.F
I.	3	1	1
II.	2	1.5	1
III.	3	1.2	1.3
IV.	2	2	2
V.	2.1	2	3

Minimalizáljuk a teljes munka hosszát!

16. Öt munkafeladatot kell szétosztani négy munkás között úgy, hogy egyik munkás sem végezheti a sorszámának megfelelő feladatot, valamint az 1. számú munkát mindenképpen el kell végezni. Egy munkás csak egy munkafeladatot végezhet el. Minimalizáljuk az összmunkaórák számát, ha az egyes munkások a következő időt töltenék az egyes feladatokkal!

Munkások	Munkák				
	1.	2.	3.	4.	5.
A	6	5	7	9	10
B	4	6	8	8	7
C	5	4	7	7	8
D	9	5	7	4	8

17. Az IBUSZ-hoz egyszerre öt külföldi turistacsoport érkezik, éspedig angol, bolgár, ciprusi, dán és egyiptomi. Az egyes csoportok az előbbi sorrendnek megfelelően rendre a következő nyelveket beszélik: angol, bolgár, görög, dán es arab. A csoportok kalauzolására a következő öt idegenvezető áll rendelkezésre: Pető Péter, Rónai Rezső, Sebők Sarolta, Tatai Tiborné, Vari Vilmosné. Ezek közül Pető Péter mind az öt nyelvet beszéli, míg a többiek 3-3 idegen nyelvet beszélnek (de nem azonos szinten). A nyelvtudás nagymértékben befolyásolja az idegenvezetés színvonalat. Az alábbi táblázat 0-tól 10-ig terjedő számokkal mutatja azt, hogy az idegenvezetők milyen színvonalon beszélik az egyes nyelveket (A 10-es a legjobb nyelvtudást jelenti, míg a 0-s azt mutatja, hogy az illető idegenvezető a nyelvet egyáltalán nem beszéli.)

Idegenvezetők	Nyelvek				
	angol	bolgár	görög	dán	arab
P	9	7	4	6	6
R	10	5	8	0	0
S	6	9	0	0	8
T	9	7	0	5	0
V	10	0	8	0	8

Az egyes csoportok kalauzolását melyik idegenvezetőre bízza az IBUSZ, ha az a célja, hogy az idegenvezetés átlagos szintje maximális legyen? (Természetesen egy idegenvezető csak egy turistacsoportot kalauzolhat.)

18. Öt hallgató vizsgázik. A tanár kiválaszt 7 tételt, amelyekből a hallgatók véletlenszerűen kihúznak egyet-egyet. Mégis tudjuk, hogy az A hallgató az első három tételből húz, az I. tételt biztos kihúzza valaki míg a VII.-et senki sem. A hallgatók felkészültséget az egyes tételekből (osztályzatokban kifejezve) a következő táblázat mutatja:

Hallgató	Tételek						
	I.	II.	III.	IV.	V.	VI.	VII.
A	3	4	4	2	5	2	5
B	4	3	3	1	2	1	3
C	5	5	2	3	2	3	4
D	4	2	3	3	2	1	3
E	3	2	3	1	3	1	3

A legszerencsésebb húzás esetén mennyi lesz a hallgatók osztályzatának átlaga? (Legszerencsésebb húzásnak azt tekintjük, ha a hallgatók által elért összeredmény a lehető legnagyobb.)

Mi a helyzet, ha azt is tudjuk, hogy biztosan születik 2-es osztályzat?

19. Egy Zala megyei építőipari vállalat jelenleg hat építkezésen dolgozik. Az építőipari vállalat dolgozói a megye négy településéről járnak munkahelyeikre. A vállalat hozzájárul dolgozói utazási költségeihez. Az utazási költségekhez való hozzájárulás attól függ, hogy a dolgozó honnan melyik munkahelyre jár dolgozni. A vállalat csökkenteni szeretné a hozzájárulás összegét. A szakmunkások

átcsoportosításánál nagyon kicsiny a lehetőség. Viszont egy-egy segédmunkás bármelyik munkahelyre irányítható, ott teljes értékű munkát tud végezni, függetlenül az építkezés készültségi fokától. Az egyes építkezéseken rendre 15, 18, 27, 13, 40, 8 fő segéderőre van szükség, míg az egyes településeken a vállalat rendelkezésére áll rendre 30, 30, 20, 28 fő segédmunkaerő. Látható, hogy a rendelkezésre álló segédmunkaerő kevesebb annál, mint amennyire szükség lenne. Ezért az építkezések igényét teljes mértékben nem lehet kielégíteni. A vállalat illetékesei - egyéb szempontokat is mérlegelve - úgy döntöttek, hogy a 4. es 6. munkahely igényét mindenképpen ki kell elégíteni, míg a többi építkezésen is biztosítani kell az igényelt segédmunkaerőnek legalább a 80%-át.

Ezeket a feltételeket figyelembe véve határozza meg, hogy az egyes településekről hova, hány segédmunkást irányítsanak, ha cél az utazási hozzájárulás minimalizálása, és az egy személy utazásához való (fajlagos) hozzájárulást az egyes utaknak megfelelően a következő táblázat mutatja:

	M1	M2	M3	M4	M5	M6
I.	520	440	320	440	500	600
II.	520	180	220	390	390	390
III.	700	610	390	610	680	490
IV.	510	440	170	290	450	410

(A táblázat számai Ft/fő dimenzióban adottak.)

Mennyi lesz a hozzájárulási összköltség? Csökkenne-e a hozzájárulási összköltség, ha a 4. es a 6. munkahelyekkel kapcsolatban is csak azt a kikötést tennék, hogy az igényelt munkaerőnek legalább a 80%-ot biztosítani kell?

20. Öt olajmezőről négy olajfinomítót látnak el nyersanyaggal. Az olajmezők és az olajfinomítók távolságai km-ekben kifejezve az alábbiak:

	F1	F2	F3	F4
M1	320	80	70	300
M2	200	50	150	100
M3	40	280	180	70
M4	150	200	500	180
M5	300	300	580	400

Az olajmezők napi termelése rendre 6000, 1000, 500, 3700, 2000 tonna. A finomítók napi kapacitása 2000, 2200, 3000 és 6000 tonna. A szállítás vasúton történik, kivéve az M2 -> F4 viszonylatban, ahol olcsóbb a közúti szállítás, de a tartálykocsik száma korlátozott, napi 200 tonna termék szállítását teszi lehetővé. Ugyanakkor az M4 -> F1 vasútvonal túlzsúfoltsága miatt legfeljebb napi 500 tonna kőolaj szállítását vállalja.

a) Hogyan tervezzük meg a finomítók nyersanyagellátását, hogy a szállítások tonnakilométere minimális legyen?

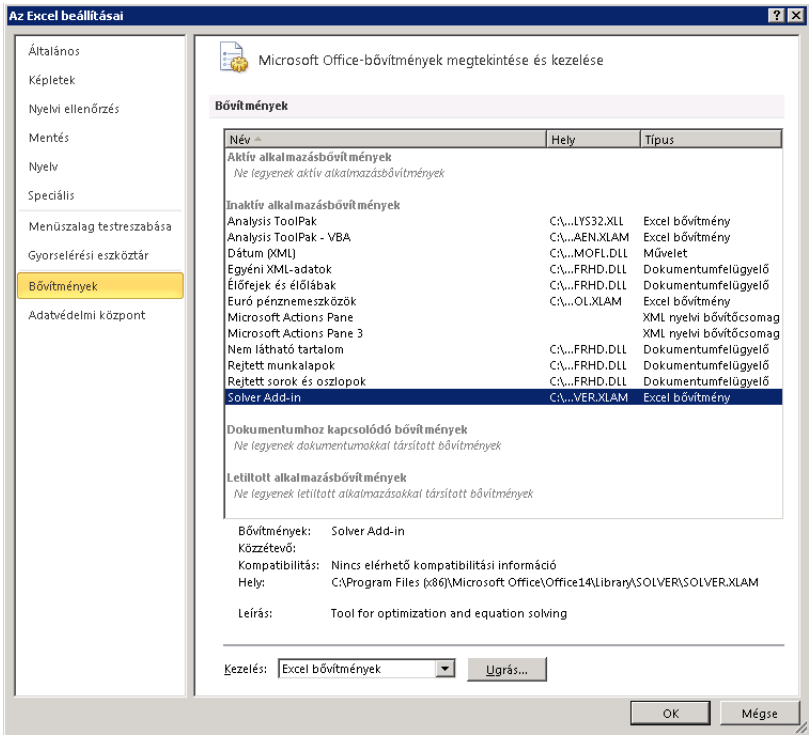
b) Hány %-kal csökken a tonnakilométer mennyisége, ha a korlátozások megszűnnek?

A. függelék

Telepítési útmutatók

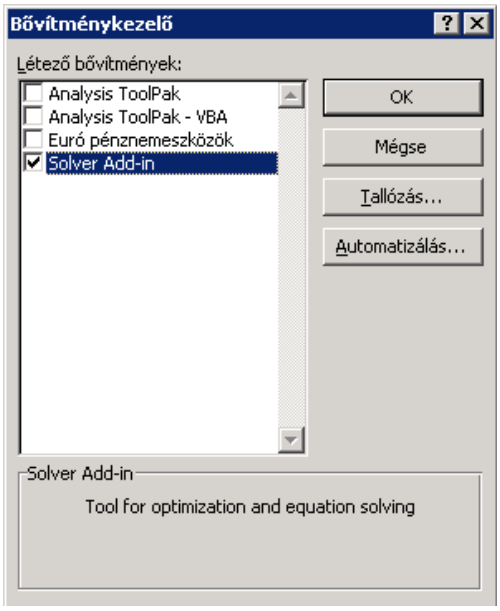
A.1. Excel solver telepítése

Az Excel Solver telepítéséhez nincs szükség telepítőlemezre, ez az Office telepítésével a merevlemezre került, csak nem lett aktiválva. Az aktiváláshoz az Excel Fájl menüjében válasszuk ki a Beállítások menüjét, ekkor az A.1. ábrán mutatott ablakot kapjuk,



A.1. ábra. Az Excel beállításai, bővítmények menü.

ahol a Bővítmények menüben a „Solver Add-in” Excel bővítményt kell kiválasztanunk. Ekkor az ablak alján az „Ugrás” gombra kattintva kapjuk az A.2. ábrán látható Bővítménykezelő ablakot.



A.2. ábra. Bővítménykezelő ablak.

A Solver melletti négyzetet kipipálva az „OK” gomb megnyomásával a program bemásolja a megfelelő állományt a helyi könyvtárba, és így az Adatok fül jobbszélén meg fog jelenni a „Solver” gomb, ahogy az a 4.9. ábrán láthattuk. Nem rendszergazda jogosultságokkal rendelkező felhasználó esetén csak felhasználószintű lesz a telepítés, vagyis minden felhasználónak magának kell elvégeznie a fenti lépéseket. Ez azt is jelenti, hogy a saját könyvtárunkba kerül a Solver állománya, így ha az méretét tekintve korlátozott, akkor biztosítanunk kell, hogy legyen hely a bővítménynek.

A.2. AMPL telepítése

Az AMPL telepítéséhez szükségünk lesz a telepítő fájlokra. Ezeket beszerezhetjük az AMPL honlapjáról, www.ampl.com, vagy a hallgatott kurzus honlapjáról, mint például az Optimalizálási Modellek kurzus honlapja. Ha az AMPL honlapjáról töltjük le a fájlokat, akkor linuxos operációs rendszer esetén ez azzal a kellemetlenséggel jár, hogy a megoldókat külön-külön kell letölteni, és ez nem is mindig egyszerű (pl. cplex). Ezért javasoljuk inkább a hallgatott kurzus honlapját a letöltéshez. A letöltött tömörített állományt mindjárt oda tömörítsük ki, ahova menteni szeretnénk, ugyanis külön installáló nem adott hozzá. A kitömörítés után csak annyi a dolgunk, hogy

lehetővé tegyük a futtatható fájlok elindítását más könyvtárakból is. Ez linux esetén azt jelenti, hogy ha a `/home/user/AMPL` könyvtárba csomagoltuk ki a fájlokat, akkor a `~/.bashrc` fájl végére írjuk be az `export PATH=$PATH:~/AMPL` sort (például a `joe` vagy verb|nano| konzolos szerkesztővel). Mentés után parancssorból hajtsuk is végre a `source ~/.bashrc` paranccsal (további belépéskor már erre nem lesz szükség). Ha mindent jól csináltunk, akkor a terminálban (parancssorban) az `ampl` parancs az `ampl:` választ adja.

Windowsos installálás esetén a futtatandó fájl az `sw.exe` ami egy parancssort ad tulajdonképpen, innen indíthatjuk az `ampl`-t.

A.3. GAMS telepítése

A GAMS telepítéséhez töltsük le az operációs rendszerünknek megfelelő fájlt a számítógépünkre a www.gams.com/download oldalról. Amíg a Windowsos telepítés egyszerű, Linuxos rendszer esetén ez már nem mondható el. A letöltött fájlt futtathatóvá téve (`chmod +x linux*sfx.exe`) indítsuk el a telepítőt abból a könyvtárból, ahova telepíteni szeretnénk. Természetesen olyan könyvtárt válasszunk, amihez van jogosultságunk. Például ha az `/usr/gams/` könyvtár lesz a helye (ehhez rendszergazdai jogosultság kell, felhasználóival a `~/gams` célszerűbb), akkor vagy másoljuk oda a telepítőt, és úgy futtassuk onnan, vagy csak lépjünk be a `cd /usr/gams` paranccsal (létrehozni a `mkdir /usr/gams`-sal tudjuk) és a telepítő teljes elérési útját megadva indítsuk el, pl. `/tmp/linux_x86_32_sfx.exe`, amennyiben a `/tmp` volt a letöltési könyvtár. Ezzel tulajdonképpen még csak kicsomagoljuk a fájlokat a `/usr/gams/gamsXX_linux_x86_32_sfx` könyvtárba (XX az aktuális verziószám), a telepítés csak ezután következik. A létrehozott könyvtárba belépve a `./gamsinst` paranccsal indítjuk el az installálást. Először az LP feladatok megoldóját kell kiválasztanunk. Mi a 6. fejezetben az XPRESS megoldót használtuk, de bármelyiket választhatjuk a sorszám megadásával. Ezután a többi feladattípushoz is kiválasztjuk a megoldókat, az alapértelmezett kiválasztásához üssünk egyszerűen „Entert”. Az összes megoldó kiválasztása után az utolsó kérdést kapjuk, ami arra vonatkozik, hogy ki férhet hozzá majd az installált fájlokhoz. Ez igazán rendszergazdai telepítésnél érdekes, ekkor érdemes mindenki számára hozzáférhetővé tenni a fájlokat. Most már csak a futtatható fájlok helyét kell az elérési úthoz hozzáadni, hasonlóan az AMPL telepítésénél leírt parancsokkal. A telepítéssel így készen is volnánk. Mivel nem adtunk meg sehol licensz fájlt, így minden demo módban fog működni, azaz korlátozott méretű feladatokat tudunk csak megoldani, de ez nekünk elegendő lesz.

Irodalomjegyzék

[1] Leo Liberti. *Problems and exercises in Operations Research*, École Polytechnique, 2007.