Projekt dolgozat
Hozzárendeléses feladat modellezése Excel-ben (VBA)

Hallgatók:
Károlyi Krisztián, Kristó Csongor, Mezei Róbert

Csíkszereda
2023

Tartalomjegyzék

Hozzárendelési feladat modellezése Excel-ben

Kivonat: *A dolgozatunk egy olyan hozzárendelési problémára segít optimális megoldást találni lineáris programozási módszerrel, ahol n tanuló vizsgázik, m db tétel közül húzhatnak egyet, és mindegyik tanulónak ismerjük az adott tételre való felkészültségét, amelyek jegyben (1 – 10) vannak megadva (n, m és a jegyek a felhasználó által tetszőlegesen módosíthatóak. Erre egy interaktív, felhasználóbarát programot készítettünk Excel-ben, VBA nyelv segítségével.*
A probléma fő kérdései pedig, amelyekre megoldást ad a program, a következők:

- *Mennyi lesz a legjobb esetben a tanulók átlaga? Ez milyen húzási kombináció mellett valósul meg?*
- *A fenti esetben lehetséges-e, hogy minden egyes tanuló azt a tételt húzza, amelyikből a legjobban felkészült?*
- *A legrosszabb átlagot adó esetben van-e olyan tanuló, aki 4-est húz?*

Emellett egy érzékenységi vizsgálatot és végeztünk, amikor is a felhasználó adhatja meg a módosítandó paraméter cellatartományát, a módosítandó érték korlátjait és a léptéket. A célfüggvény változását pedig grafikonnal szemléltetjük.
Végül a tesztelésre is nagy hangsúlyt fektetünk, hisz enélkül egyetlen program sem adható át a megrendelőnek.

Kulcsszavak: *operációkutatás, hozzárendelési feladat, excel, vba, lineáris programozás, optimalizáció*

JEL kód: *https://www.aeaweb.org/econlit/jelCodes.php?view=jel*

C44, C61, C72, C53

1. Bevezetés

1.1. A kiindulási feladat

Négy hallgató vizsgázik. A tanár kiválaszt 6 tételt, amelyekből a hallgatók véletlenszerűen kiválasztanak egyet-egyet. A hallgatók felkészültségét az egyes tételekből az alábbi táblázat tartalmazza:

1. táblázat – az alapfeladatot szemléltető táblázat

Hallgató	Tétel					
	I.	II.	III.	IV.	V.	VI.
A	9	6	9	5	7	9
B	6	7	7	4	5	4
C	9	9	5	7	5	10
D	8	5	8	7	5	4

A legszerencsésebb húzás esetén mennyi lesz a hallgatók osztályzatának átlaga? Legszerencsésebb húzásnak azt tekintjük, ha a hallgatók által elért összeredmény a lehető legnagyobb. Lehetséges-e, hogy minden hallgató azt a tételt húzza, amelyből a legjobban felkészült? A legszerencsétlenebb húzás esetén előfordulhat-e, hogy egyik hallgató sem kap 4-est?

Az 1. táblázat szemlélteti, hogy a probléma felírható egy egyszerű, n x m-es méretű mátrixként, amely a hozzárendelési feladatok alapja. Ha a feltett kérdésekre magunktól, számítógépes segítség nélkül keresnénk a választ, az a mátrix méretétől függően rengeteg időbe és energiába kerülhet, ezért sokkal célszerűbb egy számítógépes szoftvert használni, például az Excelben található Solver modul is alkalmas erre a célra. Viszont, ha az egyén nem jártas annak használatában, vagy nem szeretne időt eltölteni azzal, hogy felírja a feladatot egy Solver által megoldható problémaként, akkor jól jöhet egy olyan segédprogram például, amit mi készítettünk.

1.2. Hozzárendelési feladatok

A hozzárendelési feladat egy alapvető kombinatorikai optimalizációs feladat. A legáltalánosabb formájában, a probléma következőképpen néz ki:
Egy adott környezetben (pl. munkahely) n db dolgozó és m darab elvégzendő feladat van. Ismerjük azt a táblázatot, amely megadja, hogy az egyes dolgozók az adott feladatot milyen költség mellett (vagy mennyi idő alatt, stb.) tudják elvégezni, ezt költségmátrixnak is nevezik.

Általában (hacsak a feladat erre nem tér ki külön) egy munkakörhöz egy dolgozó és egy dolgozóhoz egy munkakör rendelhető. A cél az, hogy mindenkinek úgy osszunk ki egy feladatot, hogy az a lehető legkisebb összköltséggel járjon (vagy legkevesebb idő alatt legyen elvégezve stb.), vagy akár maximalizálási feladat is lehet.

A megoldás során a költségmátrix mellé létrehozunk egy ugyanekkora súlymátrixot (vagy együttható mátrixot), ahol bármely elem 0 vagy 1 lehet. Ezzel tudjuk jelezni, hogy az adott munkás elvégzi-e (1), vagy nem végzi el (0) az adott munkát, mivel a célfüggvény (összköltség) egy szorzatösszeg lesz: a költségmátrix minden elemét beszorozzuk a hozzá tartozó együttható mátrix elemével, így az el nem vállalt feladatok nulla értéket adnak hozzá a célfüggvényhez.

Esetünkben a költségek valójában a jegyeket jelentik, és az, hogy minimalizálni vagy maximalizálni kell-e a célfüggvényt, az az adott kérdésen múlik (legjobb húzás, legrosszabb húzás).

2. táblázat – együttható mátrix

Hallgató	**Tétel**					
	I.	**II.**	**III.**	**IV.**	**V.**	**VI.**
A	1	0	0	0	0	0
B	0	1	0	0	0	0
C	0	0	0	0	0	1
D	0	0	1	0	0	0

A 2. táblázatban például az látható, hogy ebben a megoldásban A hallgató az első tételt, B hallgató a másodikat, C hallgató a harmadikat, és végül a D hallgató a hatodik tételt kellene húzza ahhoz, hogy a legjobb átlagot érjék el, a feltételek teljesítése mellett.

2. Elméleti háttér

3. Gyakorlati tanulmány (alkalmazás bemutatása)

3.1. Rendszerkövetelmények

Az alkalmazás Microsoft Excel-ben lett készítve, egy makróbarát (.xlsm) fájlban, emiatt a futtatáshoz elengedhetetlen a Microsoft Office programcsomag megléte a számítógépen. Emellett fontos, hogy engedélyezve legyenek a makrók, illetve a külső forrásból származó állományokat ne tiltsa le az Office, mivel ebben az esetben nem lehet megnyitni a fájlt.

3.2. VBA

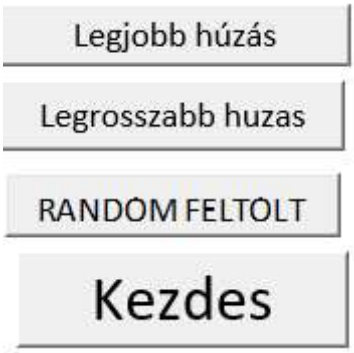
“A Visual Basic for Applications (VBA) az egyik megvalósítása a Microsoft esemény-vezérelt programnyelvének, a Visual Basic 6-nak – aminek 2008-ban szűnt meg a fejlesztése – és a

hozzá tartozó integrált fejlesztői környezetnek. Bár a Microsoft a Visual Basic támogatását és fejlesztését abbahagyta, a VBA programnyelv újabb verzióját hozták létre 2010-ben, így a Visual Basic for Applications 7 került a Microsoft Office alkalmazásokba.

A Visual Basic for Applications lehetővé teszi felhasználó által definiált függvények létrehozását, folyamatok automatizálását, és a Windows API és más alacsonyabb szintű funkciók DLL-eken keresztüli elérését. Felváltja ill. kibővíti az alkalmazások korábbi saját makró-programozási nyelveinek funkcióit, mint pl. a Word WordBASIC-éit. A futtató alkalmazás sok funkcióját lehet vele vezérelni, többek között a felhasználói felület elemeit, mint a menük és eszköztárak, és tetszőleges űrlapokat és párbeszédpaneleket. “ (forrás: Wikipédia)

3.3. A program használata

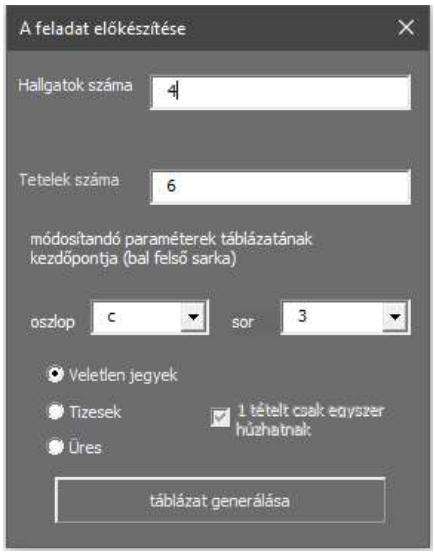
A programot igyekeztünk minél inkább könnyen kezelhetően elkészíteni. A fájl megnyitását követően máris látható a „feladat” nevű munkalap, ahol minden cella le van védve, így azok nem módosíthatóak, kivéve azon cellákat, amelyekbe adatokat kell bevigyünk.
Az A oszlopban több gomb is található, ezekkel érhetőek el a részfeladatok megoldásai is.



1. Ábra: Vezérlő gombok

A fájl (program) megnyitása után, mielőtt a „Legjobb húzás” vagy „Legrosszabb húzás” gombra kattintunk, fontos, hogy beállítsuk a tanulók számát és a tételek számát, különben hibát kaphatunk.

Egy új feladat megoldásához először a „Kezdés” gombra kell kattintani, amely megnyit egy párbeszédablakot:



2. Ábra: A feladat előkészítése

Az 1. ábrán látható ablak beviteli mezőiben van lehetőségünk megadni a tanulók számát, vagyis n-t (ennyi sor lesz a táblázatban), a tételek számát, vagyis m-et (ennyi oszlop lesz a táblázatban). Emellett be lehet állítani azt is, hogy a munkalapon belül hova generálja a program a költségmátrixot, amely elemeit mi tudjuk tetszőlegesen megadni (a táblázat bal felső celláját kell megadni, oszlop és sorszám). A munkalap első két sora és oszlopa a program működése céljából le van foglalva és nem használható erre a célra, ezért a létrehozott táblázat bal felső sarka legfeljebb a C3 cellában lehet.

A program nem enged hibás adatokat bevonni sem az Excel cellákba, sem a párbeszédablak beviteli mezőibe.

Végül ki kell választani, hogy a létrehozott táblázatban véletlen jegyek (4-10), tízesek, vagy pedig semmilyen érték se szerepeljen.

Amennyiben legalább annyi tétel van, mint hallgató (n <= m), be lehet állítani, hogy egy tételt csak egy hallgató húzhasson, ez nagyon fontos a megoldás során.

„Táblázat generálása” gombot követően máris elkészült a táblázat és szükség esetén átírhatjuk a költségmátrix elemeit, hisz a táblázat kivételt képez a munkalap védelem alól.

Ezzel együtt pedig az együtthatók mátrixa is létrejön, 4 sorral (fejlécet beleszámítva) a jegyeket tartalmazó táblázat alatt. Ezt nem tudjuk módosítani, mivel ezt a Solver algoritmusa módosíthatja csak a megoldás során. Alapból csupa egyessel van feltöltve. Mindez látható a 2. ábrán.

	Tétel 1	Tétel 2	Tétel 3	Tétel 4	Tétel 5	Tétel 6
Hallgató 1	7	3	8	10	1	6
Hallgató 2	7	6	9	10	6	10
Hallgató 3	8	6	8	5	3	10
Hallgató 4	7	4	2	4	5	1

9.25 2.75

Legjobb Atlag Legrosszabb Atlag

	Tétel 1	Tétel 2	Tétel 3	Tétel 4	Tétel 5	Tétel 6	Osszesen		
Hallgató 1	1	1	1	1	1	1	6	=	1
Hallgató 2	1	1	1	1	1	1	6	=	1
Hallgató 3	1	1	1	1	1	1	6	=	1
Hallgató 4	1	1	1	1	1	1	6	=	1
	4	4	4	4	4	4			
	<=	<=	<=	<=	<=	<=			36.5
	1	1	1	1	1	1			

3. Ábra: a táblázat generálása utáni állapot

Ezenfelül a 2. ábrán az is látszik, hogy egy Excel függvény megmutatja, hogy mekkora lenne a legjobb átlag abban az esetben, ha mindenki a neki legjobban kedvező tételt húzná (sorok maximum értékeinek az átlaga), illetve az ellenkező eset átlaga is látható, természetesen nem ezek jelentik majd a végső megoldást.

A hozzárendelési feladat egy lineáris programozási módszerrel van megoldva , így a második, általunk manuálisan nem módosítható táblázat sor összege kiszámítja automatikusan, és ezekhez egyesével hozzárendeli azt a megszorítást, hogy bármely sorösszeg pontosan 1 kell legyen, hisz minden tanuló pontosan 1 tételt húzhat. Amennyiben be lett jelölve, hogy 1 tételt csak 1 tanuló húzhat, akkor az oszlop összegekre is beszúr egy megszorítást, vagyis bármely oszlopösszeg legfeljebb 1 lehet.

A sárga háttérszínnel kiemelt szám pedig magát a megoldást adja meg, ez a Solver célfüggvénye.

3.4. Legjobb húzás átlaga (algoritmus)

Arra a kérdésre, hogy mennyi lenne a hallgatók átlaga a legjobb húzás esetén, a “Legjobb húzás” gomb ad választ. Egyúttal azt is megválaszolja egy üzenetben, hogy a megoldásban mindenki azt a tételt húzta-e, amiből a legjobb jegyet kaphatta, vagy pedig más javára le kellett mondjon az adott tételről más javára azért, hogy a csoport átlaga jobb legyen.

A megoldás során egyszerűen a gombhoz kötött alprogram beállítja a Solver paramétereit: célfüggvény cellája, módosítandó cellatartomány, maximalizálás, amit az optimális megoldás során változtat, illetve a feltételeket (Természetesen csak akkor használ korlátot a tételeknél, ha az be volt jelölve a 2. ábrán lévő jelölőnégyzet).

A megoldás után pedig rögtön megnézi azt is, hogy lehetséges-e az, hogy mindenki a neki legkedvezőbb tételt húzza-e, ezt pedig egyszerűen úgy határozza meg, hogy összehasonlítja a

sormaximumok átlagát a célfüggvény értékével, hiszen ha megegyeznek, akkor a válasz igen, különben nem.

3.5.A kihúzott tételek celláinak kiemelése

A solver lefutása után egyből kiemeli azokat a cellákat (jegyeket) a költségmátrix soraiban, amelyeket valóban kihúztak a tanulók az optimális megoldás során. Ez segít értelmezni a felhasználónak azt, hogy ki melyik tételt kell húzza ahhoz, hogy a legkedvezőbb kombinációt kapjuk, a feladathoz mérten.

Ezt úgy dönti el az algoritmus, hogy eltárolja egy-egy tömbbe a két táblázat értékeit, és két egymásba ágyazott for ciklussal megnézi páronként, hogy az adott számpáros szorzata nagyobb-e nullánál.

```
Sub Kiemel()
    TombokFeltolt
    For i = 0 To Sheet3.n
        For j = 0 To Sheet3.m
            If egyutthatok(i, j) * jegyektomb(i, j) > 0 Then
                Cells(Sheet3.startrow + i, Sheet3.startcol + j).Interior.Color = vbGreen
            Else
                  Cells(Sheet3.startrow + i, Sheet3.startcol + j).Interior.Color = vbWhite
            End If
        Next
    Next
End Sub
```

4. Ábra: A kihúzott tételek jegyeinek kiemelése

4. Érzékenységvizsgálat

5. Tesztelés

6. Következtetések

7. Könyvészet

[1] Bártfai B., *Makróhasználat Excelben*, BBS INFO kiadó, Budapest, 2010.
[2] Duţa L.D., Fábián Cs., *Metode matematice în optimizarea croirii,* Ed. Tehnică, București, 1983.
[3] *Excel és Visual Basic*, www.linamar.hu/download.php?fid=18 letöltés ideje 2020.03.18

[4] Umetani S., Yagiura M., Ibaraki T., One dimensional cutting stock problem to minimize the number of different patterns. *European Journal of Operational Research* 146 (2003) 388–402.

8. Mellékletek