Multidiszciplináris tudományok, 3. kötet. (2013) 1. sz. pp. 153-162.

DISZKRÉT FIREFLY ALGORITMUS ALKALMAZÁSI LEHETŐSÉGÉNEK VIZSGÁLATA A BESZÁLLÍTÓK KIVÁLASZTÁSÁNÁL

Kota László[1], Jármai Károly[2]
[1]tudományos segédmunkatárs, [2]egyetemi tanár
Miskolci Egyetem, Anyagmozgatási és Logisztikai Tanszék
3515 Miskolc, Miskolc-Egyetemváros, e-mail: altkota@uni-miskolc.hu

Összefoglalás

A beszállítók kiválasztása a logisztika egyik igen fontos problémaköre. A cikkben egy a firefly algoritmuson alapuló optimalizálási módszert mutatunk be, amely segít a beszállítók megválasztásában adott termék, adott igényelt mennyiség esetén. A kifejlesztett algoritmus figyelembe veszi a beszállítóknál adott minimális és maximális rendelési mennyiségeket, mint korlátokat. Valamint figyelembe veszi a nagy tételek esetén elérhető mennyiségi kedvezményeket, amely lépcsős függvénnyel írható le. Az algoritmus figyelembe veszi az alkalmazott szállítójárművek kapacitását és azok költségét is. A cikk bemutatja az algoritmus működését, az alkalmazott büntetőfüggvényeket. A cikk utolsó részében a firefly algoritmus által adott megoldást összehasonlítjuk az MS Exel nemlineáris általános redukált gradiens (ÁRG) és evolúciós megoldásával

Kulcsszavak: *beszállítók kiválasztása, optimalizálás, firefly algoritmus*

Abstract

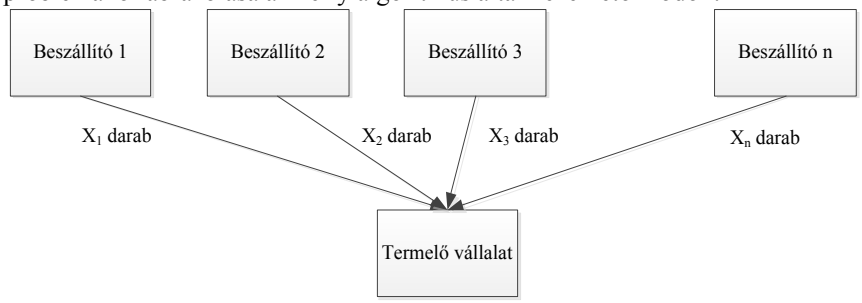
In this article we show a firefly optimization based algorithm which helps to choose the appropriate suppliers in a case of given order quantity of a given product. The developed algorithm takes account of the minimum and maximum order quantities at the different suppliers as constraints. It also takes account of the quantity discounts offered by the different suppliers, which can be described as step function. The algorithm takes account of the capacity and the cost of the used transport vehicles too. The article describes the operation of the algorithm and the penalty functions applied. In the last part the firefly algorithm and the solution given by the MS Excel solver's general reduced gradient (GRG) and the evolutionary algorithm is compared.

Keywords: *supplier selection, optimization, firefly algorithm*

1. A probléma

A beszállítók kiválasztása a logisztika egyik igen fontos problémaköre [1], különösen a just in time beszállítási rendszert alkalmazó termelő vállalatoknál [2]. A cikkben azzal a problémával foglalkozunk mikor adott beszállítóktól egyféle terméket, alkatrészt rendelünk, (*1.* ábra) viszont a beszállítók kapacitása véges így általában nem képes egyetlen beszállító az adott termékből a teljes mennyiséget beszállítani, más beszállítók bevonása is szükséges. A beszállítóknál minimális rendelési mennyiség is adott, amely alatt a beszállító gyártási

kapacitása nem vehető igénybe. A modell további bővítése, egyéb feltételek bevonása, mint t a többféle termék, gyűjtő, elosztó, körjáratok későbbi kutatások tárgyát képezik, jelen kutatás elsődleges célja az algoritmus alkalmazhatósága diszkrét állapottérben, valamint az adott problémakör ábrázolása a firefly algoritmus által kezelhető módon.



1. ábra *Beszállítók kiválasztásának problémája*

Adott a termékből, alkatrészből szükséges mennyiség: Q. Valamint adottak az egyes beszállítók által szállítható minimális és maximális mennyiségek:

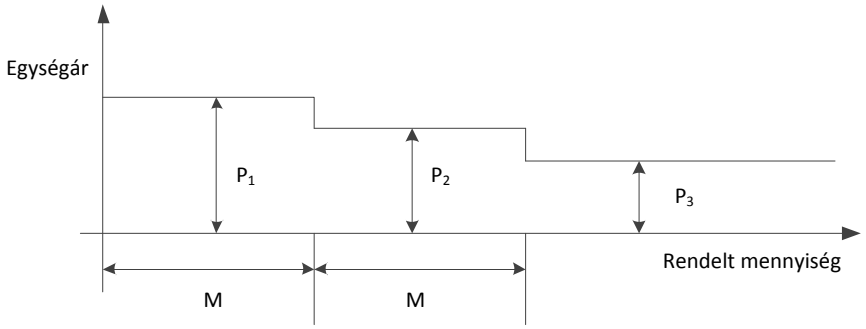
$$O_i^{min}|_{i=1..n},\ O_i^{max}|_{i=1..n}\ \text{[db]}. \tag{1}$$

Adott az egyes beszállítóktól a beszállítást végző járművek kapacitása: P_i, amelyet esetünkben konstansnak tekintünk az adott beszállítónál minden járműre. Valamint definiált az egy járműre meghatározott szállítási költség: tr_i.
A teljes szállítási költség megadható a:

$$C_i^{TR} = RoundUp\left(\frac{X_i}{P_i}\right) \cdot tr_i \quad [HUF] \tag{2}$$

összefüggéssel, ahol a RoundUp a felfelé kerekítés függvénye.

A beszerzett termék ára viszont általában a vállalatok közötti rendelések esetén függ a tételnagyságtól, hiszen nagy tételek esetén gyakran mennyiségi kedvezmények vehetők igénybe.



2. ábra *Az egységár és a rendelt mennyiség kapcsolata*

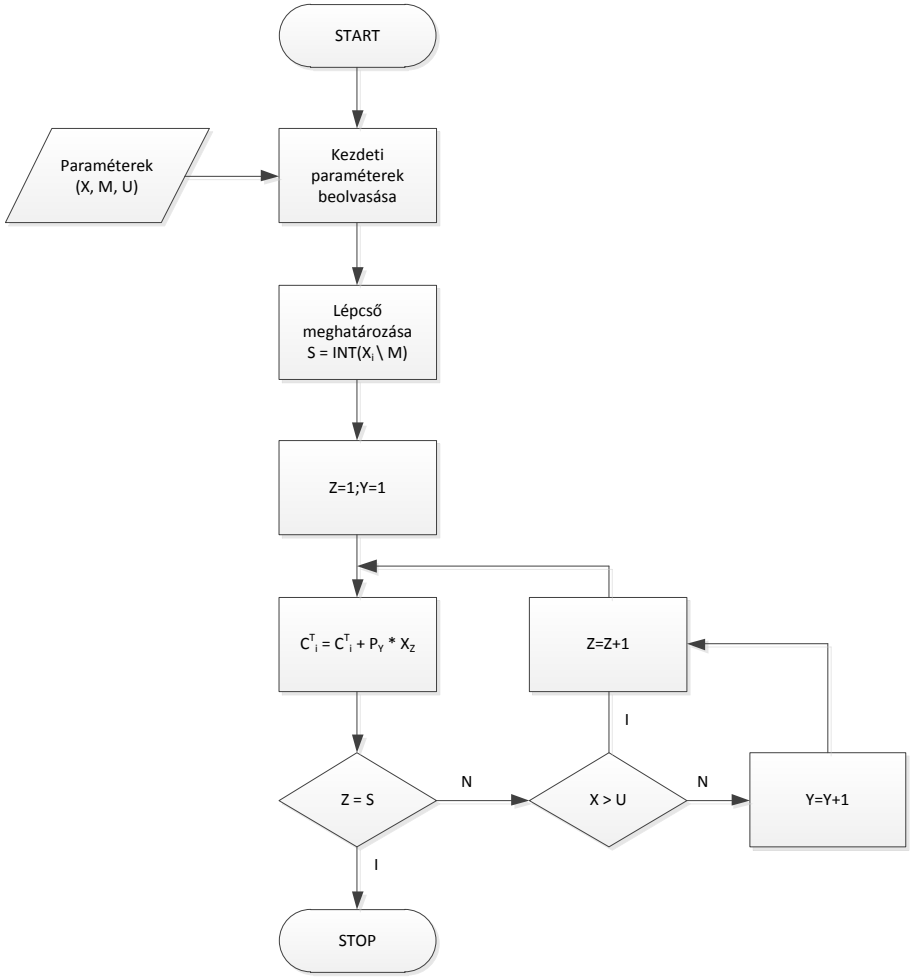
Ahogy azt a *2*. ábra mutatja, az egységár egy lépcsős függvénnyel írható le, ahol:

- M : az adott egységárhoz tartozó mennyiségi lépcső, amelyet konstansnak tekintünk, tehát a mennyiségi lépcsők mérete azonos,
- P_j : az egyes mennyiségi lépcsőkhöz tartozó egységár.

Természetesen a kedvezménnyel csökkentett egységár csak az adott mennyiség felett rendelt termékekre vonatkozik.
Esetünkben a problémát a következő modellel írjuk le:

- Minden beszállítóhoz meg kell adni az egységár kezdeti értékét: P_1 .
- Definiálni kell, hogy az egységár, mekkora mennyiség után csökken: M.
- Definiálni kell, hogy az egységár mennyit csökken az egyes lépcsők elérésével, modellünkben konstans.
- Valamint meg kell adni, hogy az egységár csökkenés hány lépésig tarthat: *U*. A modellben ez a paraméter biztosítja, hogy a kedvezmény nem vehető végtelen hosszan igénybe, tehát a termék ára a nullát nem érheti el.

A termék vásárlási költségének (C_i^T) meghatározása nem írható le egzakt módon. A költség lépcsős függvényének kiszámítása alábbi folyamatábrával jellemezhető (***3***. ábra):



3.ábra *A termék vásárlási költségének kiszámítása*

1.1 Korlátozó feltételek

A beszállítóktól vásárolt termék mennyisége nem lépheti át az adott beszállító által meghatározott minimális, illetve maximális rendelési mennyiséget:

$$O_i^{min} < X_{Ii} < O_i^{max}. \text{ [db]} \tag{3}$$

A rendelt mennyiségek összegének pontosan meg kell egyezni a szükséges mennyiséggel:

$$sum(X_i\ ) = Q|_{i=1..n} \tag{4}$$

A megrendelt mennyiség értéke csak egész szám lehet:

$$\forall X_i\ = Integer. \tag{5}$$

Az optimalizálás célfüggvénye:

$$C = \sum_{i=1}^{n}(C_i^T + C_i^{TR}) \rightarrow min. \text{ [HUF]} \tag{6}$$

Jelen egyszerűsített modellben tárolási költséget nem vesszük figyelembe, a beszállítás just in time történik.

2. FireFly algoritmus

A firefly algoritmus egy Xin-She Yang által kifejlesztett, a szentjánosbogarak szociális viselkedésén alapuló metaheurisztikus algoritmus [3]. A firefly algoritmus effektivitása összemérhető vagy jobb, mint a legújabb metaheurisztikus algoritmusok, mint például a harmony search [4], vagy a PSO módszeren alapuló új algoritmusok. A firefly algoritmusnál a szentjánosbogarak repülés közben fényjelekkel vonzzák oda a többi szentjánosbogarat. Az algoritmusban definiált mesterséges szentjánosbogarak:

- Uniszexek, egy adott szentjánosbogár fényével az összes többi szentjánosbogarat vonzza.
- A szentjánosbogarak attraktivitása egyenesen arányos a fényességükkel, két szentjánosbogár közül a fényesebb vonzza magához a kevésbé fényeset, viszont a fényesség a távolsággal csökken.
- Ha nincs az adott szentjánosbogárnál fényesebb egyed, az adott szentjánosbogár véletlenszerűen mozog.
- A szentjánosbogarak fényessége a célfüggvényen alapul.

A firefly algoritmus pszeudó kódja:

```
1. célfüggvény: f(x); X=(x1, x2, …. xd)
2. kezdeti szentjánosbogár populáció generálása: xi, (i=1….n)
3. szentjánosbogár fényesség intenzitásának (I) és célfüggvény
   kapcsolatának meghatározása I∝ f(x), vagy egyszerűen I=f(x)
4. abszorpciós koefficiens meghatározása: γ
while (t < maxgeneration)
   for i=1:n (összes szentjánosbogár)
         for j = 1:n (összes szentjánosbogár)
               if (Ij > Ii)
                     i szentjánosbogár a j felé mozog
```

```
            endif
        attraktivitás meghatározása a távolság (r) alapján
        exp(-γ·r)
        az új megoldások kiértékelése, fényintenzitások
        kiszámítása
        end for
    end for
    legjobb szentjánosbogár meghatározása
    end while
```

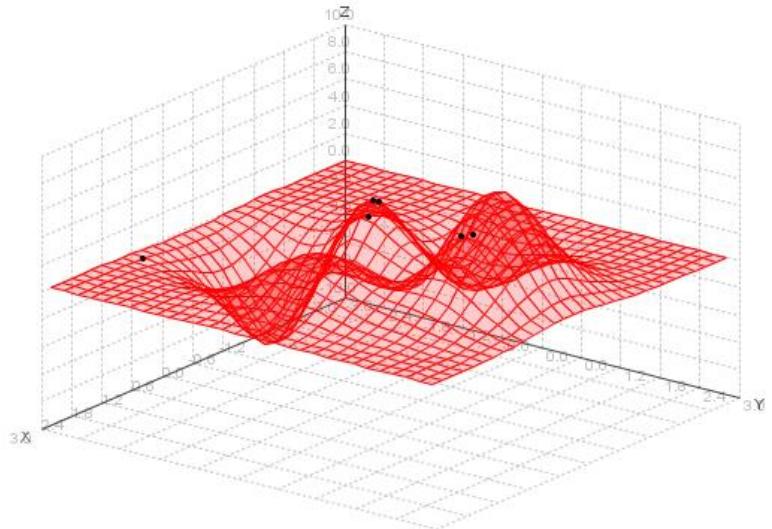
Az abszorpciós koefficiens (γ) határozza meg hogy a szentjánosbogarak fénye mennyire nyelődik el ha $\gamma \to 0$, akkor az algoritmus a normál PSO [7] (Particle Swarm Optimization), részecskeraj algoritmushoz tart.
Az egyedek mozgását legtöbbször a

$$x_{i+1} = x_i + \beta_o e^{-\gamma r_{ij}^2}\left(x_j - x_i\right) + \alpha(rand() - \tfrac{1}{2}), \tag{7}$$

vagy az ezzel egyenértékű

$$\begin{gathered} \beta = \beta_0 \cdot e^{-\gamma r} \\ x_{i+1} = x_i \cdot (1 - \beta\ \ ) + x_i \cdot \beta\ \ + \alpha(rand() - \tfrac{1}{2}), \end{gathered} \tag{8}$$

formulával határozzák meg.



4. ábra *FireFly algoritmus maximumkeresése a Peaks függvényen, xyz valós számtérben*

A firefly algoritmus eredetileg folyamatos problémák megoldására lett kifejlesztve (*4*. ábra), de az algoritmus diszkretizálható, így használható permutációs problémák megoldására is [5].

3. A probléma megoldása

A beszállítói probléma megoldása a firefly algoritmus diszkretizálásával valósul meg. Az egyes szentjánosbogarak egy-egy megoldást reprezentálnak, hasonlóan a genetikus algoritmusnál alkalmazott kromoszómákhoz. Az egyedek jóságának meghatározásához az algoritmus büntetőfüggvényeket használ.

A szentjánosbogarak kezdeti populációjának előállításakor a (4) feltételnek megfelelő egyedeket generálunk.

FireFly 1	X_1	X_2	X_3		X_d
FireFly 2	X_1	X_2	X_3		X_d
.	.	.	.	.	.
FireFly n	X_1	X_2	X_3		X_d

5. ábra *Szentjánosbogarak populációja, adatstruktúra*

A következő lépésben minden szentjánosbogárra meghatározzuk annak fényességét. A fényesség a célfüggvény (6) és a büntetőértékek összegéből adódik.

3.1 Az alkalmazott büntetőfüggvények

Minimum rendelési mennyiség alatti mennyiség büntetése:

$$B^{min} = \sum_{j=1}^{d} \left(X_j^{min} - X_j\ \right)^2 \mid if\ X_j\ < X_j^{min} \tag{9}$$

ahol:

- d: a beszállítók száma,
- X_j^{min} a j-edik beszállítótól rendelhető minimális mennyiség.

Maximális rendelési mennyiség feletti mennyiség büntetése:

$$B^{max} = \sum_{j=1}^{d} \left(X_j\ - X_j^{max}\right)^2 \mid if\ X_j\ > X_j^{max} \tag{10}$$

ahol:

- d: a beszállítók száma,
- X_j^{max} a j-edik beszállítótól rendelhető minimális mennyiség.

Maximális rendelési mennyiség feletti mennyiség büntetése:

$$I = C + B^{min} + B^{max}. \tag{11}$$

Majd következő lépésben az egyes egyedek a fényesebb egyed felé mozdulnak el. A legfényesebb egyed pedig véletlenszerűen mozog.

A mozgás meghatározása folyamatos térben egyértelmű, azonban diszkrét egyedeknél definiálni kell az egyedek távolságát valamint a léptetést is.

Az egyedek távolságának meghatározása a következő függvénnyel történik:

$$D(F1, F2) = \sum_{j=1}^{d} abs(X_j^{F1} - X_j^{F2}) \tag{12}$$

ahol:

- F1: a távolságfüggvény első operandusa, egy tetszőleges szentjánosbogár,
- F2: a távolságfüggvény második operandusa, egy tetszőleges szentjánosbogár,
- X_j^{F1}: a j-edik beszállítótól rendelt mennyiség az első szentjánosbogárnál,
- X_j^{F2}: a j-edik beszállítótól rendelt mennyiség a második szentjánosbogárnál.

Az egyedek mozgása a firefly algoritmusnál folyamatos mozgásfüggvény [6] diszkretizált variánsa.

Mivel a célfüggvény komponensek összege adott véges érték, így két adott szentjánosbogárnál csak páros számú eltérések fordulhatnak elő (*6. ábra*), jelölt mennyiségek). Tehát, ha egy adott rendelési mennyiség eltér, akkor annak az eltérésnek egy másik rendelési mennyiségnél fordított előjellel elő kell fordulnia.

	X_1	X_2	X_3	X_4	X_5	Total
FireFly 1	100	100	100	100	100	500
FireFly 2	100	99	100	101	100	500

6. ábra *Azonos végösszegű egyedek*

A mozgatás első lépéseként meghatározzuk a két szentjánosbogárnál a „több”(M) és „kevesebb” (L) eltéréseket.

$$M_i = \begin{cases} 0 \; if \; X_i^{F1} < X_i^{F2} \\ 1 \; if \; X_i^{F1} > X_i^{F2} \end{cases} \tag{13}$$

$$L_i = \begin{cases} 0 \; if \; X_i^{F1} > X_i^{F2} \\ 1 \; if \; X_i^{F1} < X_i^{F2} \end{cases} \tag{14}$$

Majd meghatározzuk a két szentjánosbogár távolsága (12) alapján a lépés nagyságát.

$$S = RoundUp\left(\frac{D^2}{R^2}\right) \tag{15}$$

ahol:

- R^2: a rendelhető mennyiségek tartománya, gyakorlatban: 0-tól Q-ig -> Q

Majd véletlenszerűen választunk egy „több” (M) és egy „kevesebb” (L) eltérést, ahol „több” értéket csökkentjük, a „kevesebb” értéket növeljük, így a szentjánosbogarak távolsága csökken.

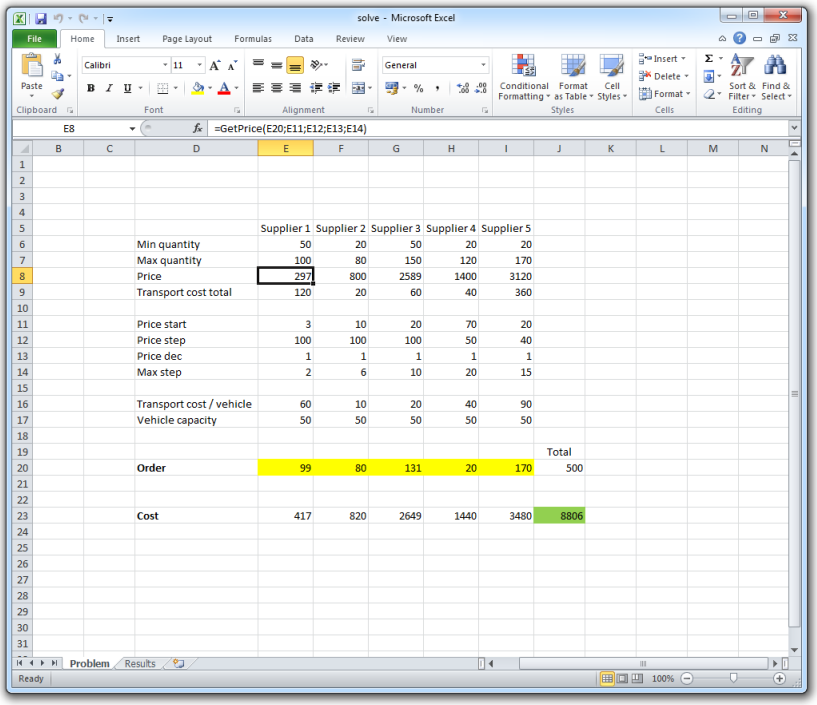
$$\begin{aligned} X_i &= X_i - S \\ X_j &= X_j + S \end{aligned} \tag{16}$$

A véletlenszerű mozgásnál, mivel mindig páros elemen kell változtatni, hogy a végösszeg azonos, maradjon, egy véletlenszerűen kiválasztott mennyiségből kivonunk, míg egy másik véletlenszerűen kiválasztott mennyiséghez hozzáadunk egy adott konstans értéket. A kísérletek azt mutatták, hogy esetünkben a legjobb konvergenciát az 1-es érték adja. Tehát a véletlenszerű mozgás leírható:

$$\begin{aligned} X_i &= X_i - 1, \\ X_j &= X_j + 1. \end{aligned} \tag{17}$$

4. Összehasonlítás az MS Excel solver megoldásával

Hasonlítsuk össze a megoldást egy példán keresztül a Microsoft Excel Solver modulja megoldásával. A példában 5 beszállítót tételezünk fel véletlenszerűen felvett értékekkel (*7. ábra*).



7. ábra *Megoldás Excel solverrel*

A probléma nem linearitása miatt az Excel solver szimplex algoritmusa már nem képes megoldani a problémát, „Az LP megoldó által igényelt linearitási feltételek nem teljesülnek” hibával leáll. Így a nemlineáris ÁRG (általános redukált gradiens módszer) és az evolúciós megoldókat vizsgáltuk meg (1. táblázat).

A költségkedvezményeket kiszámító lépcsős függvényt, valamint a szállítási költséget meghatározó függvényt VBA (Visual Basic for Applications) programnyelven publikus függvényként egy külön VBA modulban szükséges definiálni, hogy az Excel képes legyen használni az optimálás folyamán. Ilyen módon definiálva a felhasználói függvényt az a munkalapon az Excel saját függvényeihez hasonlóan használható (*7.* ábra).

Első lépésben meg kell határozni Az Excel solver kimeneti paramétereit (*8.* ábra):

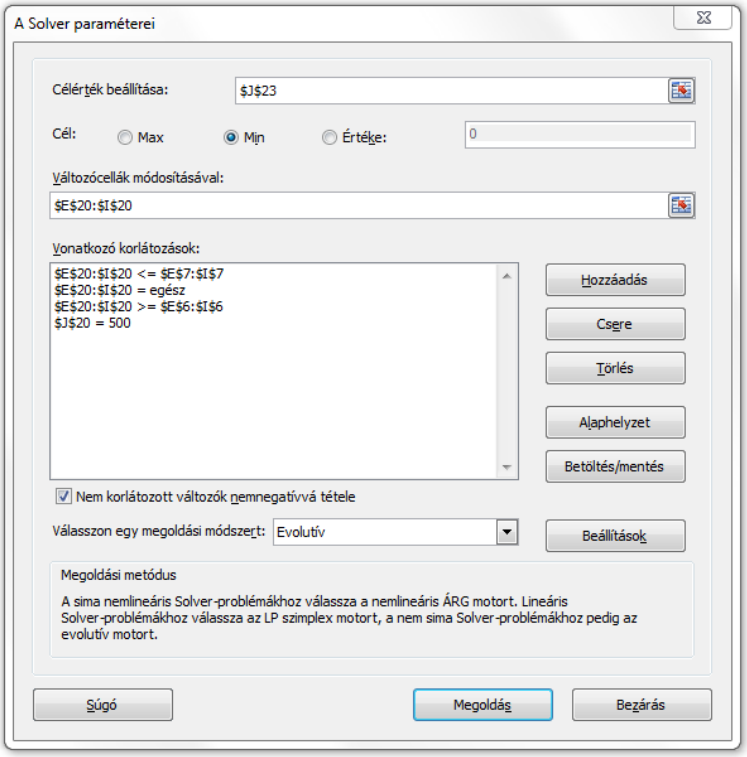
- J23: célcella, ebben a cellában számítódik ki az összes költség, amelyet az eljárás minimalizál (*C*),
- E20:I20: változó cellák, az egyes beszállítóktól való megrendelések mennyisége (X_i).

Majd a solver paraméterei között meg kell adni a az optimálás korlátozásait, ezek a:

- a megrendelések összege nem lépheti túl az igényelt mennyiséget (*Q*): J20

- az egyes mennyiségek (X_i) nem haladhatják meg a beszállító által szállítható legnagyobb mennyiséget: pl. E-I20 < E-I7,
- az egyes mennyiségek (X_i) nem lehetnek kisebbek a beszállító által szállítható legkisebb mennyiségnél: pl. E-I20 > E-I6,
- valamint a megoldásnak egész értékűnek kell lennie: E-I20 = Integer.

Minden korlátozó feltétel megadható cellák csoportos kijelölésével.



8. ábra *Excel solver paraméterei*

1. táblázat: *Futtatások eredményei*

Method	Futásidő [s]	Relatív különbség	Célfüggvény	Relatív különbség
FireFly (JAVA)	0,085	100,00%	8806	100,00%
Nemlineáris ÁRG (egykezdőpontos centrális)	0,109	128,24%	12723	69,21%
Nemlineáris ÁRG (több kezdőpontos, haladó)	48,6	57176,47%	9058	97,22%
Nemlineáris ÁRG (több kezdőpontos, centrális)	113,6	133647,06%	8873	99,24%
Evolúciós (1 ciklus)	64,974	76440,00%	9666	91,10%
Evolúciós (3 ciklus)	117,717	138490,59%	88,06	100,00%

5. Összefoglalás

A kifejlesztett firefly algoritmuson alapuló megoldás jól skálázható, az algoritmus tetszőleges számú beszállítót képes kezelni. Míg az Excel megoldásának bővítése nehézkesebb. Viszont a futtatási eredményekből látható hogy az Excel ilyen bonyolultságú probléma esetén alapértelmezésben nem képes megtalálni az optimumot, habár a firefly algoritmus futásidejének többszörösét használva jól megközelíti. A megoldó módszerek közül egyedül az evolúciós módszer találta meg a helyes eredményt többszöri futtatásra. Végül az evolúciós módszer harmadik futtatásra találta meg a helyes eredményt. Ilyenkor az előzőleg talált optimumot kiindulási pontnak használva a megoldás minden futtatáskor tovább javul. Az eredményekből látható azonban hogy a firefly algoritmus azonban igen rövid futásidővel, az evolúciós módszer idejének töredékével, szolgáltatja a helyes megoldást.

6. Köszönetnyilvánítás

A kutatás az Európai Unió és Magyarország támogatásával, az Európai Szociális Alap társfinanszírozásával a TÁMOP 4.2.4.A/2-11-1-2012-0001 azonosító számú „Nemzeti Kiválóság Program – Hazai hallgatói, illetve kutatói személyi támogatást biztosító rendszer kidolgozása és működtetése konvergencia program” című kiemelt projekt keretei között valósult meg. A kutató munka részben a Miskolci Egyetem stratégiai kutatási területén működő Innovációs Gépészeti Tervezés és Technológiák Kiválósági Központ keretében valósult meg, valamint az OTKA 75678 számú és OTKA T 109860 projektek támogatásával.

7. Irodalom

[1] Telek, P.: Operation strategies for delivery models of regional logistic networks Advanced Logistic Systems, Theory and Practice, Vol. 4., pp.: 33-40., HU ISSN 1789-2198, Miskolci Egyetem 2010

[2] Bányai, Á.: Optimisation of intermediate storage network of JIT purchasing, Advanced Logistic Systems: Theory and Practice 5: pp. 35-40. (2011), ISSN: 1789-2198

[3] Yang, X. S.: *Nature-Inspired Metaheuristic Algorithms*, Luniver Press 2008, p. 128, ISBN 1-905986-10-6

[4] Bányai T. : Optimisation of a Multi-Product Green Supply Chain Model with Harmony Search, DAAAM International Scientific Book 2011, pp. 15-30. ISBN:978-3-901509-84-1, doi: 10.2507/daaam.scibook.2011.02

[5] Kusuma G., Suyanto J.: *Evolutionary discrete firefly algorithm for travelling salesman problem*, ICAIS'11 Proceedings of the Second international conference on Adaptive and intelligent systems, Springer-Verlag Berlin, Heidelberg 2011 pp.: 393-403, ISBN: 978-3-642-23856-7

[6] Kazem Sayadia M., Ramezaniana R., Ghaffari-Nasaba N.: *A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems*, International Journal of Industrial Engineering Computations 1 (2010) 1–10, doi: 10.5267/j.ijiec.2010.01.001

[7] Farkas, J., Jármai, K.: *Design and Optimization of Metal Structures*, Horwood Publishing Limited 2008, ISBN 978-1-904275-29-9, 300 p.