TÖBBSZINTŰ OPTIMÁLÓ ALGORITMUS ALKALMAZÁSA

APPLICATION OF MULTILEVEL OPTIMIZATION ALGORITHM

Dr. Kota László [*], *Prof. Dr. Jármai Károly* [**]

ABSTRACT

In our industry researches we often face very difficult problems where ordinary algorithms fail to find the global optimum. Mostly they have difficult, high-dimensional count, very large state space where even the concept of direction and distance are non-existent and have to be defined, the neighborship in the state space also needs definition. In these cases, these terms are often defined and calculated by heuristic functions. On these problems the applied optimization methods often fail, they stuck in local optima, working very slowly and find suboptimal solution. So, we decided to try to link optimization methods and create multi-level optimization methods to cope these problems. As a base concept in the first stage we use some simple, fast, rapidly converging algorithm, then some finer grade algorithm like population-based swarm optimization method. In this paper we will show and evaluate some multi-level optimization methods tested on several test functions, comparing the convergence and computational needs.

1. BEVEZETÉS

A kutatásunk alapötlete a gyors globális és a lassabb lokális keresőalgoritmusok kombinációjának vizsgálata. Elsőként a Firefly algoritmust vizsgáltuk, mint lokális optimáló módszer, mivel ezt az algoritmust korábban többször is használtuk kutatásainkban, valamint számos publikációnk fűződik hozzá [1, 2, 3].

A Firefly algoritmus egy általános optimáló eljárás, általában gyors konvergenciával rendelkezik, számos probléma megoldására használható, ezek főleg folyamatos problémák, de különböző módosításokkal diszkrét állapottérben is használható. Gyors globális algoritmusnak első vizsgálatunk tárgyául a véletlen keresést választottuk, ami meglepő lehet, de ennek az algoritmusnak előnye a könnyű implementálhatóság mellett a gyorsaság és kis számításigény. A véletlen keresés kezdeti szakaszában nagyon gyorsan állít elő lehetséges megoldásokat, valamint gyorsan konvergál, így ideális választásnak tűnt a módszer kezdeti vizsgálatára. Vajon a Firefly algoritmus hatékonysága növelhető ezzel a módszerrel?

2. TESZTFÜGGVÉNYEK

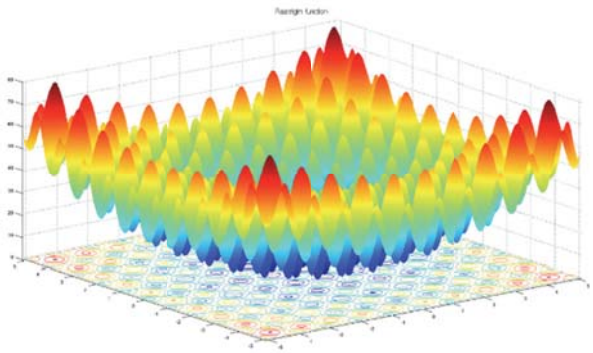
A szakirodalomban megszokott, jól bevált tesztfüggvényeket használtunk fel [4, 5]. De figyeltünk arra, hogy lehetőleg komplex sok lokális optimummal rendelkező tesztfüggvényeket használjunk a vizsgálatok folyamán (1, 2 és 3 ábra). A következő tesztfüggvényeket [6] vizsgáltuk:

Rastrigin függvény:

$$f(x) = 10d + \sum_{i=1}^{2}\left[x_i^2 - 10\cos\left(2\pi x_i\right)\right] \quad (1)$$

ahol:

d: a dimenzió.



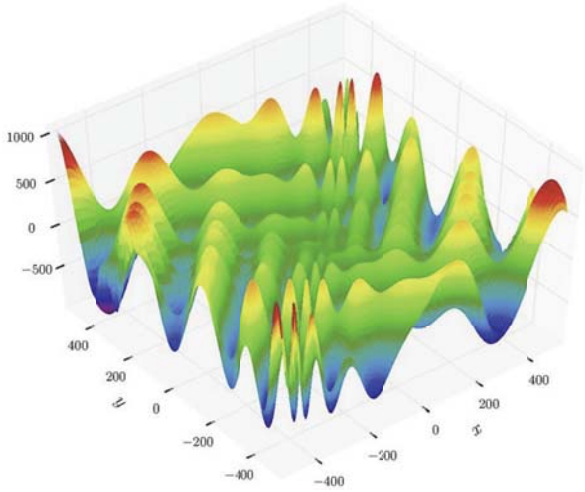
1. ábra Rastrigin függvény

Minimuma: f(0,0)=0;
Keresési tér: -5.12 ≤ x,y ≤ 5.12

Eggholder függvény:

$$f(x,y) = -(y+47)\sin\sqrt{\left|\frac{x}{2}+(y+47)\right|} - x\sin\sqrt{|x-(y+47)|} \quad (2)$$

* *egyetemi adjunktus, Miskolci Egyetem Logisztikai Intézet*
** *egyetemi tanár, Miskolci Egyetem, Energetikai és Vegyipari Gépészeti Intézet*

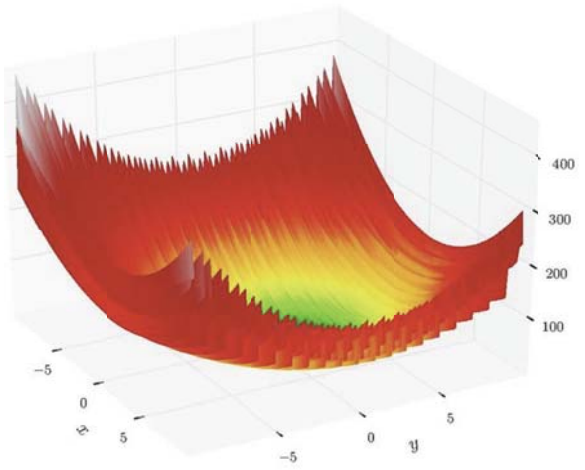


2. ábra Eggholder függvény

Minimuma: f(512,404.2319)=-959.6407;
Keresési tér:: -512 ≤ x,y ≤ 512

Lévi N.13 függvény:

$$f(x,y) = sin^2(3\pi x) + (x-1)^2\left(1 + sin^2(3\pi y)\right) + (y-1)^2\left(1 + sin^2(2\pi y)\right) \quad (3)$$



2. ábra Lévi N.13 függvény

Minimuma: f(1,1)=0;
Keresési tér: -10 ≤ x,y ≤ 10

3. AZ ALGORITMUS

Mint ahogy már említettük az első – globális – fázisban a véletlen keresés algoritmust alkalmaztuk. Első lépésben a kezdőpopulációt 100 véletlen egyeddel inicializáltuk, ami azt jelenti, hogy a célfüggvényt 100 véletlen kiválasztott helyen értékeltük ki. A vizsgálandó függvények könnyű cserélhetőségének érdekében nem csak a Firefly módszer hanem a véletlen keresés módszer is populációt használt. Program-technikailag a populáció egy konténer objektum, melynek minden eleme szintén egy objektum, ezek az elemek reprezentálják a célfüggvény egy megoldását, valamint tartalmazzák az optimáló algoritmusok számára használható egyéb adattagokat és metódusokat. Ezzel a módszerrel a populációt használó algoritmusok, mint például a genetikus algoritmus vagy az esetünkben használt Firefly algoritmus reprezentációja könnyen megvalósítható. Esetünkben a véletlen keresés módszer is ezt a populáció objektumot használta.

A globális keresés algoritmusának pszeudó kódja:

- a populáció inicializálása
- for x=0:population.count
    - temp.x=Random; temp .y=Random
    - temp.Fitness = CalculateTargetFuction(x,y)
    - temp.Fitness < population[x].Fitness -> population[x]=temp
- endfor

A második fázisban a lokális keresési algoritmus, esetünkben a Firefly algoritmus kapja meg a populációt bementként. A lokális keresés algoritmusának pszeudó kódja:

- for i=0:population(firefly).count
- for j=0:population(firefly).count
    - if firefly[i].Fitness < firefly[j].Fitness -> MoveToward firefly[j]->firefly[i]
- endfor j
- endfor i

- for i=0:population(firefly).count
    - if firefly[i] not moved -> MoveRandom firefly[i]
- endfor i

A Firefly algoritmusnál minden mozgás az állapottérben egy függvénykiértékelést jelent.

4. TESZTFUTTATÁSOK

Rastrigin függvény teszfuttatás: ahogy a kapott adatok is mutatják jelentős javulást értünk el a függvény kiértékelések számának csökkentése terén. A második futtatást vettük fel referenciapontnak, ahol a Firefly algoritmust egyedül futtatva az optimumérték már nem javult az iterációk számának növelésével. A futtatások során a két optimáló eljárás futás számának arányát, majd magát a futások számát is változtatva az eredeti függvény kiértékelés szám 6 százalékára csökkentettük a szükséges kiértékelési számot, valamint az optimum pontossága is jelentősen javult (1. táblázat).

1. táblázat Rastrigin függvény futtatás (RndS – Véletlen keresés, FF – Firefly algoritmus)

	Iterációszám						
Futás	RndS	FF	Fitnesz	X	Y	Függvény kiértékelések száma	Kiértékelések száma százalék
1	0	200	0,01352	0,00686	0,00459	943947	195,64%
2	0	100	0,01352	0,00686	0,00459	482501	100,00%
3	0	50	0,02213	0,00425	-0,00967	243250	50,41%
4	50	50	0,00013	-0,00063	-0,00050	248925	51,59%
5	50	10	0,00058	0,00002	-0,00170	52288	10,84%
6	50	5	0,00242	0,00307	0,00168	29171	6,05%

2. táblázat Eggholder függvény futtatás (RndS – Véletlen keresés, FF – Firefly algoritmus)

	Iterációszám						
Futás	RndS	FF	Fitnesz	X	Y	Függvény kiértékelések száma	Kiértékelések száma százalék
1	0	200	-959,606	512	404,4073	1006508	200,76%
2	0	100	-959,528	512	403,9173	501347	100,00%
3	0	50	-959,511	512	403,8936	249682	49,80%
4	50	50	-959,639	512	404,2674	248491	49,56%
5	50	10	-959,639	512	404,2674	53805	10,73%
6	50	5	-959,639	512	404,2674	29712	5,93%

3. táblázat Lévi N.13 függvény futtatás (RndS – Véletlen keresés, FF – Firefly algoritmus)

	Iterációszám						
Futás	RndS	FF	Fitnesz	X	Y	Függvény kiértékelések száma	Kiértékelések száma százalék
1	0	200	8,67E-06	1,000122	1,002706	958042	196,28%
2	0	100	3,32E-05	1,000595	0,998815	488090	100,00%
3	0	50	3,32E-05	1,000595	0,998815	245813	50,36%
4	50	50	3,67E-06	1,000201	1,000173	250824	51,39%
5	50	10	6,32E-06	1,000116	1,002262	54921	11,25%
6	50	5	0,000281	0,998402	1,007171	29901	6,13%

Eggholder függvény tesztfuttatás: A második futtatási sorozatnál az Eggholder függvényt használtuk, az eredmények javulása hasonló az első esethez, a relatív értékek egy kicsit még jobbak is annál. Ebben az esetben a függvény kiértékelések száma az eredeti függvény kiértékelés szám 6 százaléka alá süllyedt (5,93%). Az összehasonlíthatóság miatt itt is ugyanazt az iterációszámot használtuk, mint az első esetben, habár az optimum javulás nem állt meg a második futtatásnál a Firefly algoritmus esetében a tendencia hasonló. Az Eggholder függvény esetén is nemcsak az iterációszám csökkent drámaian, hanem az optimálás pontossága is javult ezzel párhuzamosan (2. táblázat).

Lévi N.13 függvény tesztfuttatás: Végül, de nem utolsósorban a Lévi N.13 függvényt vizsgáltuk. Ebben az esetben az optimálás függvény kiértékelései számának javulása nem olyan jó mint az előző esetben, de nemsokkal tér el tőle. Itt is az eredeti függvény kiértékelés szám majdnem 6 százalékára (6,13%) sikerült csökkenteni a kiértékelések számát habár ebben az esetben az optimum pontossága 5-7 tízezredet romlott. Az 5-ös esetben, ahol a pontosság még javult a relatív kiértékelés száma 11,25 százalékos, ami szintén egy nagyságrend javulás az eredeti értékhez képest (3. táblázat).

5. KÖVETKEZTETÉSEK

Ahogy a cikkünk is mutatja, lehet létjogosultsága a többfázisú algoritmusoknak, jól megválasztott lokális globális algoritmuspárokkal. Egy gyors globális keresőalgoritmus sokat segíthet egy lokális vagy egy általános keresőalgoritmusnak, ami drámaian kihat a számításigényre gyorsabb kiszámítást vagy kevesebb processzoridőt biztosítva. Az így nyert számítási kapacitást bonyolult problémáknál igen jelentős lehet. Jelenlegi fázisban még nem jelenthetjük ki a módszer általános alkalmazhatóságát, habár az eredmények jelentős javulást mutatnak. További futtatások, más tesztfüggvényekkel, valamint más algoritmuspárok kipróbálása, tesztelése is szükséges, mint például genetikus módszerek kombinálása véletlen kereséssel, valamint a Firefly algoritmus mellett más modern swarm módszer [7] tesztelése. Valamint szeretnénk a módszert kiterjeszteni sokdimenziós diszkrét problémákra. Hiszen sok valós probléma folyamatos függvényekkel nehezen leírható. Legtöbbször mátrixokkal, legtöbb esetben sok döntési változós állapottérben, akár a problémákra kifejlesztett metrikákkal és szomszédsági függvényekkel operálva [8, 9].

5. KÖSZÖNETNYILVÁNÍTÁS

A cikkben ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése” projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

6. IRODALOM

[1] KOTA L., JÁRMAI K.: Preliminary studies on the fixed destination mmtsp solved by discrete firefly algorithm, Advanced Logistic Systems: Theory and Practice (ISSN: 1789-2198) 7: (2) pp. 95-102. (2014)

[2] KOTA L., JÁRMAI K.: Diszkrét Firefly algoritmus alkalmazási lehetőségének vizsgálata a beszállítók kiválasztásánál, Multidiszciplináris Tudományok: a Miskolci Egyetem közleménye 3:(1) pp. 153-162. (2013)

[3] KOTA L., JÁRMAI K. Szentjánosbogár algoritmus diszkretizálása több utazó ügynökös probléma megoldására, GÉP 65:(8) pp. 21-24. (2014)

[4] JÁRMAI K., MARCSÁK G.Z., BARCSÁK C.: Application of test functions for the evaluation of metaheuristic algorithms. In: Proceedings of International Conference on Innovative Technologies, IN-TECH 2015, Dubrovnik, Croatia, 09–11 September 2015, pp. 251–255 (2015). ISSN 1849-0662

[5] MOLOGA, M., SMUTNICKI, C.: Test functions for optimization needs, pp, 1–10 (2014). http://www.robertmarks.org/Classes/ENGR5358/Papers/functions.pdf. Accessed 27 Mar 2017

[6] https://en.wikipedia.org/wiki/Test_functions_for_optimization. Last Accessed 30 Mar 2017

[7] BÁNYAI Á., BÁNYAI T, ILLÉS B.: Optimization of Consignment-Store-Based Supply Chain with Black Hole Algorithm,” Complexity, vol. 2017, Article ID 6038973, doi:10.1155/2017/6038973

[8] FARKAS, J., JÁRMAI, K.: Optimum Design of Steel Structures. Springer, Heidelberg (2013). 288 p. ISBN 978-3-642-36867-7. http://dx.doi.org/10.1007/978-3-642-36868-4

[9] KOTA, L., JÁRMAI, K.: Mathematical modelling of multiple tour multiple traveling salesman problem using evolutionary programming. Appl. Math. Model. 39(12), 3410–3433 (2015). doi:10.1016/j.apm.2014.11.043