Diferencialna evolucija za večkriterijsko optimizacijo s samoprilagajanjem in z lokalnim preiskovanjem SQP

Aleš Zamuda, Janez Brest, Borko Bošković, Viljem Žumer
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Smetanova ul. 17, 2000 Maribor, Slovenija
ales.zamuda@uni-mb.si

Differential Evolution for Multiobjective Optimization with Self-adaptation and SQP Local Search

This paper presents a hybrid algorithm for multiobjective optimization which combines our self-adaptive differential evolution algorithm SA-DE [3] and a local search. The used local search method is implemented by an algorithm of sequential quadratic programming SQP [9]. SQP and SA-DE methods are applied to the DEMO [12] algorithm to create new candidate solutions in optimization process. Hybridization of both methods is done by improving some candidate solutions generated by differential evolution with SQP. The solution obtained with SQP replaces the candidate solution only if it dominates it. Performance assessment of the hybrid algorithm shows that the new algorithm using local search gives significantly better results than the algorithm without the local search and also improves its strength compared to some algorithms from literature.

1 Uvod

V članku predstavljamo hibridni algoritem za večkriterijsko optimizacijo. Kombinirali smo samoprilagodljivo diferencialno evolucijo SA-DE [3] za globalno preiskovanje in sekvenčno kvadratično programiranje SQP [1] za lokalno preiskovanje. Ta način preiskovanja smo uporabili pri izdelavi novih kandidatnih rešitev v optimizacijskem procesu algoritma DEMO [12]. Hibridizacijo diferencialne evolucije in lokalnega preiskovanja smo izvedli tako, da SQP skuša izboljšati nekatere kandidatne rešitve iz DE. Rešitev iz SQP zamenja staro kandidatno rešitev le, če jo dominira. Kot je videti iz ocene kakovosti dobljenega algoritma, novi algoritem z lokalnim preiskovanjem daje signifikantno boljše rezultate kot algoritem brez lokalnega preiskovanja.

V drugem poglavju predstavimo sorodna dela. V tretjem poglavju opišemo predlagan algoritem za večkriterijsko optimizacijo, ki za preiskovanje uporablja diferenciano evolucijo in lokalno preiskovanje. V četrtem poglavju izvedemo eksperimente in opišemo rezultate. V petem poglavju podamo zaključek in predloge za nadaljnje delo.

2 Sorodna dela

2.1 Večkriterijska optimizacija

Za lažje razumevanje, kaj je večkriterijska optimizacija (ang. *Multiobjective Optimization* – MO), definirajmo njen cilj. Cilj MO je reševanje večkriterijskega optimizacijskega problema (ang. *Multiobjective Optimization Problem* – MOP). MOP definiramo kot iskanje dopustnega vektorja spremenljivk oziroma iskalnih parametrov $\mathbf{x} = (x_1, x_2, ..., x_D)$, ki optimizira (v članku bomo na optimizacijo gledali kot minimizacijo) vektorsko funkcijo $\mathbf{f}(\mathbf{x})$:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_M(\mathbf{x})).$$

Z ovrednotenjem vektorja $\mathbf{x}$ iz prostora spremenljivk $\Re^D$ dobimo kriterijski vektor $\mathbf{f}(\mathbf{x})$ v prostoru kriterijev $\Re^M$. Vektor $\mathbf{x}$ mora zadoščati m omejitvam:

$$g_i(\mathbf{x}) \geq 0; \quad i = 1, 2, ..., m,$$

s katerimi podamo enostavne omejitve, kot npr. definicijsko območje in zaloga vrednosti optimizirane funkcije. Rešitev večkriterijskega optimizacijskega problema $\mathbf{x}$, dominira rešitev $\mathbf{y}$ ($\mathbf{x} \preceq \mathbf{y}$), če velja:

1. rešitev $\mathbf{x}$ ni slabša od rešitve $\mathbf{y}$ po vseh kriterijih $f_i(\mathbf{x}) \leq f_i(\mathbf{y}), \forall i = 1, ..., M$ in
2. rešitev $\mathbf{x}$ je boljša od rešitve $\mathbf{y}$ po vsaj enem kriteriju $\exists j \in \{1, ..., M\}$: $f_j(\mathbf{x}) < f_j(\mathbf{y})$.

Pareto optimalna rešitev je takšna rešitev $\mathbf{x}$, ki je ne dominira nobena druga dopustna rešitev $\mathbf{z}$ in bi veljalo $\mathbf{f}(\mathbf{z}) \preceq \mathbf{f}(\mathbf{x})$. Množica nedominiranih rešitev v množici rešitev P je množica vseh tistih rešitev, ki jih ne dominira nobena rešitev iz množice P. Kriterijski vektorji rešitev iz množice nedominiranih rešitev tvorijo nedominirano fronto. Nalogo večkriterijskega optimiranja lahko tako pretvorimo v nalogo iskanja množice nedominiranih rešitev, ki jo tu imenujemo aproksimacijska množica (ang. *approximation set*).

2.2 Diferencialna evolucija

Diferencialna evolucija (DE) [14, 17] je algoritem, ki se uspešno uporablja za globalno optimizacijo realno kodiranih numeričnih funkcij. Algoritem ima malo parametrov (F – skalirni faktor diferenčnega vektorja za mutacijo, CR – stopnja križanja, NP – velikost populacije ter včasih tudi s – strategija mutacije in križanja), vendar zaradi svoje narave prilagajanja problemu in stabilnosti iskanja z elitističnim selekcijskim mehanizmom, daje boljše rezultate od ostalih evolucijskih algoritmov [2, 10].

Algoritem diferencialne evolucije [14] sestoji iz glavne evolucijske zanke, v kateri z evolucijskimi operatorji mutacije, križanja in selekcije postopno in vzporedno izboljšuje približek iskane rešitve. Evolucijski operatorji vplivajo na vsak primerek $\mathbf{x}_i$, $\forall i \in [0, NP]$ v populaciji rešitev, iz katerih se zgradi nova populacija za naslednjo generacijo (ang. *generation*). Eno kreiranje novega osebka imenujemo iteracija (ang. *iteration*). V vsaki iteraciji operator mutacije izračuna mutiran vektor (ang. *donor vector*) $\mathbf{v}_{i,G+1}$:

$$\mathbf{v}_{i,G+1} = \mathbf{x}_{r_1,G} + F \times (\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}),$$

kjer so $r_1, r_2, r_3 \in 1, 2, ..., NP$ paroma in od i različni indeksi primerkov iz populacije v generaciji G, $i \in 1, 2, ..., NP$ in $F \in [0, 2]$. Vektor r_1 imenujemo osnovni vektor (ang. *base vector*). Izraz $\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}$ imenujemo diferenčni vektor (ang. *difference vector*) in po množenju s faktorjem ojačanja F, utežen diferenčni vektor (ang. *weighted difference vector*).

Po mutaciji dobljeni mutiran vektor $\mathbf{v}_{i,G+1}$ križamo s ciljnim vektorjem (ang. *target vector*) $\mathbf{x}_{i,G}$ in tako dobimo poskusni vektor (ang. *trial vector*) $\mathbf{u}_{i,G+1}$. Operator križanja v algoritmu DE prevzema dve obliki, imenovani binarno križanje ('/bin') ali eksponentno križanje ('/exp'). Prvo zapišemo kot:

$$u_{i,j,G+1} = \begin{cases} v_{i,j,G+1} & \text{rand}(0,1) \leq CR \text{ ali } j = j_{rand} \\ x_{i,j,G} & \text{sicer} \end{cases},$$

kjer $j \in [1, D]$ označuje j-ti iskalni parameter v prostoru z D dimenzijami, funkcija $\text{rand}(0, 1) \in [0, 1]$ označuje vzorčenje uniformno (psevdo) naključno porazdeljenega naključnega števila in j_{rand} izbira uniformno naključen indeks iskalnega parametra, ki ga vedno izmenjamo (da bi s tem preprečili izdelavo enakih posameznikov). CR označuje že omenjen krmilni parameter stopnje križanja.

Enokriterijska selekcija v algoritmu DE za vsak nov generiran primerek preveri, ali je ocena $f(\mathbf{u}_{i,G+1})$ poskusnega vektorja boljša od ocene ciljnega vektorja $f(\mathbf{x}_{i,G})$:

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1} & \text{if } f(\mathbf{u}_{i,G+1}) < f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G} & \text{sicer} \end{cases}.$$

2.3 Samoprilagajanje krmilnih parametrov

Algoritem SA-DE [3] prilagaja krmilne parametre DE z rekombinacijo krmilnih parametrov vseh vektorjev, ki sodelujejo v evolucijskem procesu za izdelavo novega vektorja. Faktor ojačanja diferenčnega vektorja F_i za vsakega od posameznikov i za novo generacijo $G + 1$ izračunamo:

$$F_{i,G+1} = \langle F_G \rangle_i \times e^{\tau N(0,1)},$$

kjer τ označuje stopnjo učenja in je ponavadi sorazmeren s $\tau \sim 1/\sqrt{2D}$. $N(0, 1)$ je po Gaussu porazdeljena naključna spremenljivka. Izraz $\langle F_G \rangle_i$ označuje povprečenje parametra F:

$$\langle F_G \rangle_i = \frac{F_{i,G} + F_{r_1,G} + F_{r_2,G} + F_{r_3,G}}{4}.$$

Za krmilni parameter CR uporabimo podobni formuli:

$$CR_{i,G+1} = \langle CR_G \rangle_i \times e^{\tau N(0,1)},$$

$$\langle CR_G \rangle_i = \frac{CR_{i,G} + CR_{r_1,G} + CR_{r_2,G} + CR_{r_3,G}}{4}.$$

Mehanizem samoprilagajanja iz algoritma SA-DE (enokriterijska optimizacija) smo vključili v algoritem DEMOwSA [16] za večkriterijsko optimizacijo. Slednji algoritem temelji na algoritmu DEMO [12].

2.4 Sekvenčno kvadratično programiranje

Sekvenčno kvadratično programiranje (SQP) [11, 1] je nelinearna optimizacijska metoda, ki sloni na izračunavanju gradienta. Je posplošitev Newtonove metode na več dimenzij. Ko podamo začetno rešitev, metoda v vsaki nadaljnji iteraciji reši aproksimacijski model in se tako skuša približati rešitvi osnovnega problema. Osnovno kriterijsko funkcijo aproksimira z njenim kvadratičnim aproksimacijskim modelom in izračuna optimum aproksimiranega kvadratičnega modela. Optimum tega modela ni nujno tudi optimum osnovne kriterijske funkcije.

Kot pri večini optimizacijskih metod, metoda SQP ni izvedena le kot en algoritem, temveč služi le kot koncept, iz katerega so se razvili številni specifični algoritmi [1]. Algoritem za metodo SQP, ki smo ga uporabili, je algoritem FSQP-AL. Implementacijo najdemo v programskem paketu CfSQP [9][1]. Metoda SQP je zmožna reševati tudi probleme z nelinearnimi omejitvami.

3 Hibridni algoritem DEMOwSA-SQP

Hibridizacijo diferencialne evolucije v algoritmu DEMOwSA in lokalnega preiskovanja SQP smo izvedli tako, da SQP skuša izboljšati nekatere kandidatne rešitve iz DE.

[1] Uporabili smo programski paket CfSQP z akademsko licenco.

Tabela 1: Razlika vrednosti indikatorja $I_{\overline{H}}$ za algoritma NSGAII-SBX in DEMOwSA-SQP na funkcijah 1–7.

FES		1. OKA2	2. SYMPART	3. S_ZDT1	4. S_ZDT2	5. S_ZDT4	6. R_ZDT4	7. S_ZDT6
5e+5	Najboljši	6,0261e-03	−4,4310e−05	−1,3119e−04	1,2354e−02	2,0312e−03	−1,1508e−03	−1,8198e−02
	Srednji	7,4775e−03	−5,8835e−05	−1,3555e−04	5,9990e−02	6,1551e−03	−3,9596e−03	−2,5899e−02
	Najslabši	9,3048e−03	−8,9082e−05	−1,8119e−04	6,1319e−02	1,4402e−02	−8,6523e−03	1,2877e−02
	Povprečje	*7,4853e−03*‡	**−6,2419e−05**†	**−1,4087e−04**†	*5,0623e−02*‡	*6,6826e−03*‡	**−4,5442e−03**†	**−2,0690e−02**†
	Std	9,8163e−04	−1,4044e−05	−1,0796e−05	1,9519e−02	3,5106e−03	−2,4800e−03	1,0566e−02

Tabela 2: Razlika vrednosti indikatorja $I_{\overline{H}}$ za algoritma NSGAII-SBX in DEMOwSA-SQP na funkcijah 8–13 z $M = 3$.

FES		8. S_DTLZ2	9. R_DTLZ2	10. S_DTLZ3	11. WFG1	12. WFG8	13. WFG9
5e+5	Najboljši	−4,5731e−04	−4,5386e−04	4,2460e−01	2,6680e−01	−1,6476e−01	3,7326e−01
	Srednji	−1,2757e−03	−4,5386e−04	3,9110e−01	2,7426e−01	−1,6013e−01	3,6690e−01
	Najslabši	−1,7472e−03	−4,5387e−04	3,7420e−01	2,8602e−01	−1,4624e−01	3,3640e−01
	Povprečje	**−1,2237e−03**†	**−4,5386e−04**†	*3,9910e−01*‡	*2,7642e−01*‡	**−1,5943e−01**†	*3,6242e−01*‡
	Std	−3,3739e−04	−1,1830e−09	−1,7560e−02	6,7697e−03	4,3755e−03	−1,2020e−02

Najprej iščemo globalno rešitev z DE in dobimo kandidatno rešitev. Nato na vsakih $gen_{LS} = 10$ generacij z do $FEs_{LS} = 5000$ ovrednotenji v lokalnem preiskovanju SQP skušamo izboljšati do $ind_{LS} = 15\%$ (to vrednost smo dobili empirično) naključno izbranih kandidatnih rešitev iz trenutne populacije. Izbiramo le posameznike, ki jih je predhodno izboljšal že globalni algoritem, saj se s tem izognemo nepotrebnemu večkratnemu lokalnemu preiskovanju iste rešitve. Najprej izvedemo $miter_1 = 3$ iteracije SQP in če dobljena rešitev dominira kandidatno rešitev iz DE, izvedemo še $miter_2 = 10$ iteracij nad izboljšano rešitvijo. Rešitev dobljena v SQP zamenja staro kandidatno rešitev iz DE le, če jo dominira. Nov algoritem poimenujemo DEMOwSA-SQP. Pri SQP smo uporabili le omejitve za prostor spremenljivk, tj. linearni omejitvi minimuma in maksimuma za vsako spremenljivko. Konvergenčno stopnjo pri SQP smo nastavili na $\epsilon = 10^{-8}$. Stopnjo samoprilagajanja krmilnih parametrov v DE smo izbrali $\tau = 2/\sqrt{2D}$.

4 Rezultati

Za ocenitev učinkovitosti iskalnih algoritmov so bile predlagane številne metrike [6], zato je večina obstoječih študij in primerjav algoritmov zasnovanih na specifičnih testnih meritvah. Te študije so največkrat statistične, izvedene nad izbranimi testnimi problemi, ki jih algoritem skuša rešiti. Mi smo uporabili testno ogrodje iz [4], ki oceni dobljene aproksimacijske množice z več indikatorji.

V tabelah 1–3 podajamo razlike ocen indikatorja $I_{\overline{H}}$ za algoritma NSGAII-SBX [13] in DEMOwSA-SQP. Algoritem NSGAII-SBX je zasedel prvo mesto na tekmovanju CEC 2007. Oba algoritma uporabljata lokalno preiskovanje SQP. Na prikazanih indikatorjih naš algoritem bolje reši 10, NSGAII-SBX 9 testnih problemov. V tabeli 4 vidimo primerjavo ocen rezultatov pri 5e+5 ovrednotenjih za algoritem DEMOwSA-SQP in ostale algoritme iz tekmovanja CEC 2007. Za šest primerjanih algoritmov (krepko poudarjene ocene v tabeli 4) doseže algoritem DEMOwSA-SQP boljše ocene glede na algoritem iz [16]. Algoritem DEMOwSA-SQP daje zelo primerljive rezultate kot algoritem GDE3 [7], ki ima za vsako funkcijo posebej nastavljene krmilne parametre in uporablja večji arhiv. Glede na ostale primerjane algoritme daje algoritem DEMOwSA-SQP večkrat signifikantno boljše rezultate od ostalih algoritmov.

Tabela 4: Povzetek ocen rezultatov algoritmov.

Ime algoritma	Vir	Indikator $I_{\overline{H}}$		
		Boljše	Slabše	Nesignif.
GDE3	[7]	**9**	10	0
NSGAII-SBX	[13]	**10**	9	0
MO_DE	[19]	11	7	1
NSGAII-PBX	[8]	**12**	7	0
MOSaDE	[5]	**13**	6	0
MTS	[15]	**13**	6	0
MO_PSO	[18]	**13**	6	0

5 Zaključek

Predstavili smo hibridni algoritem za večkriterijsko optimizacijo, ki združuje mehanizma diferencialne evolucije in lokalnega preiskovanja. Oba koncepta sta vključena v algoritmu DEMOwSA-SQP. Prikazali smo rezultate testov predstavljenega algoritma in njihovo primerjavo. Uporabljen indikator rezultate algoritma oceni bolje kot rezultate algoritma iz [16], tako da po ocenjeni kakovosti preseže algoritem NSGAII-SBX, ki je na tekmovanju CEC 2007 zasedel prvo mesto. V nadaljnjem delu bi lahko izboljšali način izbire posameznikov iz populacije, ki jih izboljšujemo z lokalnim preiskovanjem. Ena od možnih strategij za to izbiro bi bila uporaba metrik porazdeljenosti posameznih kriterijskih vektorjev. Ostaja tudi

Tabela 3: Razlika vrednosti indikatorja $I_{\overline{H}}$ za algoritma NSGAII-SBX in DEMOwSA-SQP na funkcijah 8–13 z $M = 5$.

FES		8. S_DTLZ2	9. R_DTLZ2	10. S_DTLZ3	11. WFG1	12. WFG8	13. WFG9
5e+5	Najboljši	−1,4319e−06	−3,2240e−01	7,7460e−02	5,1837e−01	−6,1435e−01	7,8890e−02
	Srednji	−2,5238e−06	−4,2600e−01	4,8770e−02	5,2887e−01	−6,2723e−01	9,1950e−02
	Najslabši	−1,8275e−05	−4,6330e−01	3,6311e−02	5,3206e−01	−6,2627e−01	1,0891e−01
	Poprečje	**−4,3354e−06**†	**−4,2420e−01**†	*5,1580e−02*‡	*5,2726e−01*‡	**−6,2244e−01**†	*9,2560e−02*‡
	Std	−3,5624e−06	−2,7360e−02	−1,1340e−02	3,6232e−03	−4,2050e−03	5,2078e−03

možnost drugačne nastavitve krmilnih parametrov hibridizacijskega postopka algoritma DEMOwSA in lokalnega preiskovanja SQP.

Literatura

[1] P. T. Boggs in J.W. Tolle. Sequential quadratic programming for large-scale nonlinear optimization. *Journal of Computational and Applied Mathematics*, 124(1-2):123–137, 2000.

[2] J. Brest, S. Greiner, B. Bošković, M. Mernik in V. Žumer. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Transactions on Evolutionary Computation*, 10(6):646–657, 2006.

[3] J. Brest, A. Zamuda, B. Bošković in V. Žumer. An Analysis of the Control Parameters' Adaptation in DE. V Uday K Chakraborty, urednik, *Advances in Differential Evolution, Studies in Computational Intelligence*, letnik 143. Springer, 2008.

[4] V. L. Huang, A. K. Qin, K. Deb, E. Zitzler, P. N. Suganthan, J. J. Liang, M. Preuss in S. Huband. Problem Definitions for Performance Assessment & Competition on Multi-objective Optimization Algorithms. Tehnično poročilo TR-07-01, Nanyang Technological University et. al., Singapore, 2007.

[5] V. L. Huang, A. K. Qin, P. N. Suganthan in M. F. Tasgetiren. Multi-objective Optimization based on Self-adaptive Differential Evolution. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3601–3608. IEEE Press, 2007.

[6] J. Knowles, L. Thiele in E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, februar 2006. dopolnjena različica.

[7] S. Kukkonen in J. Lampinen. Performance Assessment of Generalized Differential Evolution 3 (GDE3) with a Given Set of Problems. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3593–3600. IEEE Press, 2007.

[8] A. Kumar, D. Sharma in K. Deb. A Hybrid Multi-Objective Optimization Procedure Using PCX Based NSGA-II and Sequential Quadratic Programming. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3011–3018. IEEE Press, 2007.

[9] C. T. Lawrence, J. L. Zhou in A. L. Tits. User's Guide for CFSQP Version 2.5: A C Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying All Inequality Constraints. Tehnično poročilo, Institute for Systems Research, University of Maryland, College Park, MD 20742, 1997. TR-94-16r1.

[10] E. Mezura-Montes in B. C. Lopez-Ramirez. Comparing bio-inspired algorithms in constrained optimization problems. *IEEE Congress on Evolutionary Computation (CEC 2007)*, strani 662–669, 2007.

[11] G. V. Reklaitis, A. Ravindran in K. M. Ragsdell. *Engineering Optimization: Methods and Applications*. Wiley-Interscience, 1983.

[12] T. Robič in B. Filipič. DEMO: Differential Evolution for Multiobjective Optimization. V *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization – EMO 2005*, letnik 3410 of *Lecture Notes in Computer Science*, strani 520–533. Springer, 2005.

[13] D. Sharma, A. Kumar, K. Deb in K. Sindhya. Hybridization of SBX Based NSGA-II and Sequential Quadratic Programming for Solving Multi-objective Optimization Problems. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3003–3010. IEEE Press, 2007.

[14] R. Storn in K. Price. Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[15] L. Y. Tseng in C. Chen. Multiple Trajectory Search for Multiobjective Optimization. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3609–3616. IEEE Press, 2007.

[16] A. Zamuda. Samoprilagajanje krmilnih parametrov pri algoritmu diferencialne evolucije za večkriterijsko optimizacijo. Magistrska naloga, Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru, 2008.

[17] A. Zamuda, J. Brest, B. Bošković in V. Žumer. Large Scale Global Optimization Using Differential Evolution with Self Adaptation and Cooperative Co-evolution. V *2008 IEEE World Congress on Computational Intelligence*, strani 3719–3726. IEEE Press, 2008.

[18] K. Zielinski in R. Laur. Adaptive Parameter Setting for a Multi-Objective Particle Swarm Optimization Algorithm. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3019–3026. IEEE Press, 2007.

[19] K. Zielinski in R. Laur. Differential Evolution with Adaptive Parameter Setting for Multi-Objective Optimization. V *The 2007 IEEE Congress on Evolutionary Computation CEC 2007*, strani 3585–3592. IEEE Press, 2007.