IT4Innovations
Centre of Excellence

A Modification of Adaptive Differential Evolution

Radka Poláková

Centre of Excellence IT4Innovations
Division of the University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
Ostrava, Czech Republic

ISCAMI 2013, Malenovice

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum
a vývoj pro inovace

IRAFM

Outline

Global Optimization Problem

- objective function $f : S \to \mathcal{R}$, $S \subset \mathcal{R}^D$
- $\boldsymbol{x}^*$ is called global minimum point when

$$\forall \boldsymbol{x} \in S, f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$$

- search space S is closed compact set

$$S = \prod_{i=1}^{D} [a_i, b_i]; a_i < b_i \quad i = 1, 2, \ldots, D$$

Differential Evolution Algorithm – Pseudocode

1: generate an initial generation P, $(\boldsymbol{x}_i, i = 1, 2, \ldots, NP)$
2: evaluate f in each $\boldsymbol{x}_i, i = 1, 2, \ldots, NP$
3: **while** stopping condition not achieved **do**
4: **for** $i := 1$ **to** NP **do**
5: generate a new trial vector $\boldsymbol{y}$ using P
6: evaluate f in $\boldsymbol{y}$
7: **if** $f(\boldsymbol{y}) \leq f(\boldsymbol{x}_i)$ **then** insert $\boldsymbol{y}$ into Q
8: **else** insert $\boldsymbol{x}_i$ into Q
9: **end if**
10: **end for**
11: $P := Q$
12: **end while**

DE – Generating Trial Vector

- trial vector $\boldsymbol{y}$ – generated by mutation and crossover operations
- computed from current point $\boldsymbol{x}_i$ and mutant $\boldsymbol{v}$ by a kind of crossover
- mutant $\boldsymbol{v}$ – developed by a kind of mutation from some points of current generation of population P
- many types of mutation were proposed
- used types of crossover: binomial and exponential
- DE strategy – a combination of a mutation and a crossover

Differential Evolution (DE) Algorithm

- proposed in 1997 by Storn and Price
- parameters of DE – size of population NP, type of mutation, parameter of mutation F, type of crossover, parameter of crossover CR, stopping condition
- setting of parameters can take a lot of time when particular optimization problem is solved by DE
- adaptation – effective instrument to avoid time-consuming setting of parameters
- Competitive DE – one of the most effective adaptive variants of differential evolution

Competitive DE (CDE), Tvrdík (2006)

- let us have H settings
- we choose among them randomly with probabilities q_h, $h \in \{1, 2, \ldots, H\}$
- h-th setting is successful if it generates a trial point $\boldsymbol{y}$ better then $\boldsymbol{x}_i$, $f(\boldsymbol{y}) \leq f(\boldsymbol{x}_i)$
- probabilities q_h, $h \in \{1, 2, \ldots, H\}$, are adjusted according to success rate of settings in preceding steps

$$q_h = \frac{n_h + n_0}{\sum_{j=1}^{H} (n_j + n_0)}$$

  n_h is current count of successes of the h-th setting, $n_0 > 1$ prevents a dramatic change in q_h by random successful use of h-th setting
- to avoid degeneration of the strategy-choosing process, current values q_h are reset to their starting values ($q_h = 1/H$) if any $q_h < \delta$ (δ is input parameter, $\delta > 0$)

Best Performing Version of CDE – b6e6rl

- b6e6rl is competition of 12 settings of DE algorithm

Mutation	F	Crossover	CR
randrl	0.5	bin	0 0.5 1
	0.8	exp	CR1 CR2 CR3

Aims of This Work

- to propose some modification(s) to the best performing variant of CDE algorithm b6e6rl
- to compare the new version of b6e6rl (modification(s) included) and original algorithm on hard benchmark functions defined for CEC2013

Outline

Observation of b6e6rl Behavior

- algorithm b6e6rl quickly converges to global optimum point for some functions from benchmark set
- but sometimes converges to wrong point (local optimum) (premature convergence)

Good diversity

- current generation of population P has good diversity, if its members are dispersed in whole search space S or at least in enough large part of the search space S
- enough large part of search space is $(1/2)^D$ of S in our tests

The First Tool Implemented to b6e6rl

- if current generation of population has good diversity, it is stored
- if population P consists of NP same members, this point is remembered and population is refreshed
- refreshing, the last stored generation of population (with good diversity) becomes a current generation and the search continues

Observation of b6e6rl Behavior

- following situation sometimes comes for some functions:
  $\forall \boldsymbol{x}$ from current generation of population P holds:
  $f(\boldsymbol{x}) = c$, c is constant and $\exists i$ $(1 \leq i \leq NP)$ that $\boldsymbol{x}_i \neq \boldsymbol{x}_j$, $\forall j$, $1 \leq j \leq NP$, $j \neq i$

The Second Tool Implemented to b6e6rl

- if population has NP members with the same value of objective function, trial point for randomly selected point of population is generated randomly (for the others by common differential evolution way)
- if situation in population changes (value of objective function for a point of population is different from constant c) trial point for each point of population is again computed by differential evolution way

Experiments – Comparison of Algorithms

- conditions for tests were set according to CEC2013 competition
- 28 benchmark functions
- dimension of test problems, $D = 2$, $D = 5$, $D = 10$
- size of population $NP = 20$ for $D = 2$, $NP = 30$ for $D = 5$, and $NP = 50$ for $D = 10$
- stopping condition was $(FES \geq 100000)$ and $(FES \geq 500000)$ for $D = 2$, $(FES \geq 250000)$ and $(FES \geq 1000000)$ for $D = 5$, and $(FES \geq 500000)$ and $(FES \geq 5000000)$ for $D = 10$
- 50 independent runs for each function and both algorithms (original b6e6rl and modified b6e6rl)
- each run has solution, if it equals to known solution the run is successful
- in following tables: nf – number of successful runs, mean – mean of the errors

D=2	b6e6rl/100000		modb6e6rl/100000		b6e6rl/500000		modb6e6rl/500000	
Function	nf	mean	nf	mean	nf	mean	nf	mean
1	50	0	50	0	50	0	50	0
2	50	0	50	0	50	0	50	0
3	50	0	50	0	50	0	50	0
4	50	0	50	0	50	0	50	0
5	50	0	50	0	50	0	50	0
6	50	0	50	0	49	0.0021	50	0
7	50	0	50	0	50	0	50	0
8	49	0.4001	50	0	50	0	50	0
9	50	0	50	0	50	0	50	0
10	40	0.0015	45	0.0007	41	0.0013	42	0.0015
11	50	0	49	0.0199	50	0	49	0.0009
12	42	0.1592	47	0.0597	47	0.0597	48	0.0398
13	48	0.0796	49	0.0154	46	0.1592	47	0.0137
14	26	1.6551	27	0.4787	27	0.8326	35	0.0999
15	10	3.9675	19	0.2127	8	3.9804	20	1.1457
16	39	0.0078	48	0.0082	34	0.0118	47	0.0023
17	7	1.7234	8	1.3825	10	1.6166	7	0.8763
18	10	1.4146	6	1.4832	6	1.6162	6	0.8269
19	45	0.0016	47	0.0010	44	0.0024	47	0.0006
20	19	0.0120	16	0.0132	12	0.0148	20	0.0117
21	40	36	42	6.9334	40	36	46	6.2444
22	39	4.2337	41	3.3422	44	1.4288	47	0.3506
23	31	25.2991	46	3.8980	41	12.3268	49	0.0104
24	38	7.5526	42	1.8086	39	7.4893	45	1.6899
25	23	54	25	24.5018	17	66.0560	28	15.9334
26	29	4.7240	23	0.1322	21	0.7631	30	0.4906
27	6	97.2004	7	64.9479	3	93.7136	9	29.0289
28	26	48	32	21.4386	24	52	45	6.3899

D=5	b6e6rl/250000		modb6e6rl/250000		b6e6rl/1000000		modb6e6rl/1000000	
Function	nf	mean	nf	mean	nf	mean	nf	mean
1	50	0	50	0	50	0	50	0
2	50	0	50	0	50	0	50	0
3	50	0	50	0	50	0	50	0
4	50	0	50	0	50	0	50	0
5	50	0	50	0	50	0	50	0
6	44	0.3937	47	0.1580	44	0.3931	46	0.3145
7	40	6.11E-07	50	0	32	2.69E-06	50	0
8	28	7.1362	30	6.9697	31	6.6210	23	9.1465
9	35	0.2046	37	0.1714	35	0.1987	35	0.2145
10	0	0.0341	2	0.0394	0	0.0421	1	0.0390
11	50	0	50	0	50	0	50	0
12	19	0.8358	18	0.9552	13	1.0820	11	1.0346
13	18	1.1577	13	1.7201	15	1.6848	10	1.6027
14	8	0.4355	17	0.1524	14	0.2715	12	0.5546
15	0	37.5859	1	36.2461	0	31.6466	0	39.0155
16	0	0.5495	0	0.4019	0	0.4833	0	0.4456
17	1	4.9335	1	4.9339	1	4.9335	2	4.8329
18	0	6.4578	0	5.8307	0	6.3217	0	5.4288
19	9	0.0466	9	0.0377	15	0.0372	11	0.0373
20	1	0.1142	1	0.1616	0	0.1688	0	0.1655
21	0	244	0	265.6271	0	240	0	227.1130
22	19	83.2453	25	59.5856	21	73.6113	17	90.3071
23	4	155.2007	9	140.2799	8	144.5745	7	158.8507
24	1	114.5986	0	105.2664	0	120.1137	0	98.4459
25	0	101.2292	0	101.5483	0	101.8223	0	101.6902
26	0	100.3638	0	99.6359	0	103.0618	0	85.9510
27	0	302.3977	0	294.9429	0	303.2075	0	293.9838
28	0	282	0	272.0231	0	280	0	204.7908

D=10	b6e6rl/500000		modb6e6rl/500000		b6e6rl/5000000		modb6e6rl/5000000	
Function	nf	mean	nf	mean	nf	mean	nf	mean
1	50	0	50	0	50	0	50	0
2	50	0	50	0	50	0	50	0
3	45	0.1320	43	0.1334	47	0.2540	46	0.1306
4	50	0	50	0	50	0	50	0
5	50	0	50	0	50	0	50	0
6	29	4.1212	26	4.7100	21	5.6912	35	2.9437
7	10	0.0008	10	0.0001	6	0.0018	46	6.16E-08
8	0	20.3544	0	20.3303	0	20.3311	0	20.3066
9	17	0.8624	14	0.8106	11	0.8838	10	0.9706
10	8	0.0246	2	0.0305	3	0.0224	3	0.0243
11	50	0	50	0	50	0	50	0
12	0	5.2979	0	4.8786	0	4.4427	0	4.2823
13	0	6.5589	0	5.7573	1	5.8123	0	5.0368
14	13	0.0799	17	0.0787	17	0.0724	14	0.0699
15	0	423.7382	0	435.3112	0	362.9956	0	367.3891
16	0	0.6452	0	0.6913	0	0.7268	0	0.7409
17	0	10.1224	0	10.1224	0	10.1224	1	9.9200
18	0	18.3715	0	17.9313	0	17.4978	0	16.2364
19	0	0.1830	0	0.1747	0	0.1029	0	0.0972
20	0	1.9617	0	1.9326	0	1.9223	0	1.7537
21	0	364.1590	0	364.1590	0	376.1707	0	372.1668
22	1	18.0475	1	20.6023	1	16.5546	0	20.4865
23	0	456.4104	0	420.9574	0	356.4936	0	356.8574
24	0	201.6003	0	202.1151	0	204.2142	0	178.8817
25	0	202.0179	0	201.4763	0	201.6341	0	190.4825
26	0	154.9491	0	162.6015	0	152.3965	0	138.5527
27	0	336.7312	0	330.5482	0	331.7843	0	313.1277
28	0	292	0	292	0	284	0	284

Results – Wilcoxon two-sample test, s.b. means significantly better, = means no significant difference

	D=2		D=5		D=10	
Function/MaxFES	100000	500000	250000	1000000	500000	5000000
1	=	=	=	=	=	=
2	=	=	=	=	=	=
3	=	=	=	=	=	=
4	=	=	=	=	=	=
5	=	=	=	=	=	=
6	=	=	=	=	=	s.b.
7	=	=	s.b.	s.b.	=	s.b.
8	=	=	=	=	=	=
9	=	=	=	=	=	=
10	=	=	=	=	=	=
11	=	=	=	=	=	=
12	=	=	=	=	=	=
13	=	=	=	=	=	=
14	=	=	=	=	=	=
15	s.b.	s.b.	=	=	=	=
16	s.b.	s.b.	s.b.	=	=	=
17	s.b.	s.b.	=	=	=	=
18	=	s.b.	s.b.	s.b.	=	=
19	=	=	=	=	=	=
20	=	=	=	=	=	=
21	=	=	=	=	=	=
22	=	=	=	=	=	=
23	s.b.	s.b.	=	=	=	=
24	=	=	=	s.b.	=	s.b.
25	s.b.	s.b.	=	=	=	s.b.
26	=	=	=	s.b.	=	=
27	s.b.	s.b.	=	s.b.	=	=
28	s.b.	s.b.	=	s.b.	=	=
	7x	8x	3x	6x	0x	4x

Outline

Conclusions

- a modification of b6e6rl was proposed in order to prevent from premature convergence in hard multimodal problems
- modified b6e6rl algorithm was applied to 28 benchmark functions developed for CEC2013 competition – more effective than original b6e6rl on a part of these benchmarks
- proposed tools implemented together helps b6e6rl algorithm to work better on tested benchmark functions but it needs large amount of function evaluations to show this fact

Thank You for Your Attention