COMPARISON OF MULTI-OBJECTIVE OPTIMIZATION METHODS

Martin Marek
Master Degree Programme (2), FEEC BUT
E-mail: xmarek54@stud.feec.vutbr.cz

Supervised by: Petr Kadlec
E-mail: kadlecp@feec.vutbr.cz

Abstract: This paper deals with comparison of multi-objective optimization methods. Basic properties of multi-objective optimization are explained here. Algorithms NSGA-II, MOPSO and GDE3 are briefly introduced and compared using performance metrics on several test functions.

Keywords: Multi-objective, optimization, NSGA-II, MOPSO, GDE3

1 INTRODUCTION

Multi-objective optimization (MOOP) is a process of finding set of optimal solutions (known as Pareto-optimal solutions) when objective functions of solved problem are conflicting. Pareto-optimal solutions express the trade-off between individual objectives. The decision whether one solution is better than the other or not is made using fitness functions values.

In the multi-objective optimization, there are two goals: To find a set of solutions as close as possible to the Pareto-optimal front. To find a set of solutions as diverse on the Pareto-optimal front as possible. First goal is identical with single-objective optimization task. Second one is specific for multi-objective optimization only.

2 OPTIMIZATION METHODS

In this section NSGA-II [1], MOPSO [2] and GDE3 [3] optimization methods are briefly described.

2.1 NSGA-II

Elitist Non-dominated Sorting Genetic Algorithm [1] uses non-dominated sorting [4] on combined parent and offspring population of 2*N* members. In a selection process, only *N* members are selected to form next parent population.

After selection process, new parent population is submited to crossover and mutation operations to create new offspring population. Whole process is repeated for given number of iterations.

2.2 MOPSO

Multi-objective Particle Swarm Optimization [2] simulates the movement of swarm of bees. At the beginning, agent's positions and velocities are randomly assigned. Then, non-dominated solutions are inserted to an external archive. For each agent personal best (*pbest*) and global best (*gbest*) positions [2] are assigned. In the next step, velocity vectors are computed to find a new agent's positions.

New positions can be out of decision variable's ranges, thus boundary conditions has to be applied. After this step, content of external archive and agent's personal and global best positions are updated. Whole process is repeated for given number of iterations.

2.3 GDE3

Third Version of Generalized Differential Evolution [3] creates trial vector $\vec{u}$ from vector of decision variables $\vec{x}$ according to equation (1).

$$u_{j,i,g} = x_{j,r_3,g} + F \cdot (x_{j,r_1,g} - x_{j,r_2,g}), \tag{1}$$

where j denotes j-th decision variable, i denotes i-th agent, g denotes g-th generation, r_1, r_2, r_3 are randomly chosen agent's indexes and F is scaling factor [3].

Afterwards $\vec{u}_{i,g}$ or $\vec{x}_{i,g}$ or both are selected to create new vector of decision variables $\vec{x}_{i,g+1}$ based on dominance relation. If both vectors are selected, overall number of solutions is larger then N, thus crowding comparison [1] has to be used to select most promising solutions.

3 COMPARATIVE STUDY

Efficiency of the above described methods was tested using several test functions. Comparison between optimization algorithms was based on Generational Distance (*GD*) [4], Spread (Δ) [4] and Hypervolume metrics [4]. Another considered parameter was computational complexity of algorithms.

Test functions used for algorithm comparison are: Schaffer's study (SCH), Fonseca's and Fleming's study (FON), Kursawe's study (KUR), Poloni's study (POL), Zitler-Deb-Thiele's: ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 [4][1].

3.1 SIMULATION RESULTS

Table 1 contains average values from 20 simulation runs. Values of computational time, generational distance, spread and hypervolume were taken after 200 iterations of populations with 200 individuals for each test function. It is not possible to localize true Pareto-optimal front for KUR and POL testing functions. Thus, generational distance cannot be calculated.

The algorithms were for purposes of our comparative study set as follows:

NSGA-II settings:	Probability of crossover - $P_C = 1$,
	Probability of mutation - $P_M = 0.5$,
	Size of mating pool (tournament selection) - $\|MP\| = 2$,
	Binary precision - $BP = 20$.
MOPSO settings:	Cognitive learning factor - $C_1 = 1.5$,
	Social learning factor $C_2 = 1.5$,
	Inertia weight - linearly decreasing from $W = 0.9$ to $W = 0.5$.
GDE3 settings:	Scaling factor - $F = 0.2$,
	Probability of crossover - $CR = 0.4$.

NSGA-II's computational time is quite stable for every test function. Small time variances between individual problems are caused by different number of decision variables. In MOPSO, if small number of non-dominated solutions is found, non-dominated sorting is performed on smaller population (in NSGA-II strictly $2N$) and diversity maintanance is not required which results in smaller computational effort. GDE3's computational time instability has similar reason as MOPSO's, but it is less apparent.

Generational distance of MOPSO and GDE3 for ZDT4 test function is quite large. Both algorithm converged to local Pareto-optimal front, which is main difficulty with ZDT4 test function.

Hypervolume values are diverse for each test function due to different sizes of objective space.

	SCH	FON	KUR	POL	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
Computational time (s)									
NSGA-II	14.551	14.202	14.33	14.28	16.553	15.918	16.599	14.193	14.03
MOPSO	3.521	2.803	2.134	4.753	1.629	0.8195	1.395	0.334	0.387
GDE3	11.737	10.13	9.666	8.979	6.006	5.663	6.045	6.211	6.347
Generational Distance (-)									
NSGA-II	1.65E-5	6.96E-4	-	-	1.15E-2	0.267	3.4E-3	2	0.251
MOPSO	2.43E-5	2.10E-3	-	-	1.24E-2	0.238	1.13E-2	7.64	4.8
GDE3	1.73E-5	3.14E-4	-	-	2.33E-2	0.176	1.08E-2	26.7	0.484
Spread (-)									
NSGA-II	0.139	0.110	0.191	0.109	0.513	0.588	0.577	0.662	0.704
MOPSO	0.084	0.300	0.849	0.184	1.439	1.187	1.438	1.042	1.099
GDE3	0.119	0.100	0.185	0.119	0.368	0.792	0.484	0.782	0.755
Hypervolume (-)									
NSGA-II	12.639	0.297	25.84	359.5	0.649	0.308	0.778	1.280	0.204
MOPSO	13.307	0.298	26.78	377.9	0.284	0.094	0.501	1.4E-5	0.906
GDE3	13.303	0.303	26.13	357.3	0.674	0.128	0.789	0.016	0.205

Table 1: Simulation results

4 CONCLUSION

Resulting values from simulation shows that although MOPSO is the fastest algorithm it is outperformed by other two algorithms in almost every cases and for complex test functions (ZDT) shows poor diversity maintanance. Algorithms GDE3 and NSGA-II show similar results for easier test function such as SCH, FON, KUR and POL, but results from more complex test functions (ZDT family) are in favor of NSGA-II. NSGA-II converges systematically to true Pareto-optimal front in wide spread of solutions (ZDT test functions need more time to scour vast decision variable space), but GDE3 can converge closer to true Pareto-optimal front if satisfactory number of individuals and iterations is used (SCH, FON and KUR test functions are sufficiently searched through with comparative study settings).

ACKNOWLEDGEMENT

This work has been supported by the Technology Agency of the Czech Republic, grant TA 04010457 ”Tools for synthesis of antennas and sensors”.

REFERENCES

[1] Deb, K., Pratap, A., Agarwal, S.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 2002, 6.2: 182-197.

[2] Reyes-Sierra, M., Coello, C. C.: Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International journal of computational intelligence research*, 2006, 2.3: 287-308.

[3] Kukkonen, S., Lampinen, J.: GDE3: The third evolution step of generalized differential evolution. In: *Evolutionary Computation, 2005. The 2005 IEEE Congress on*. IEEE, 2005. p. 443-450.

[4] Deb, K.: Multi-objective optimization using evolutionary algorithms. John Wiley & Sons, 2001.