Zachowania oparte o modele w heterogenicznym roju cząstek

Mateusz Zaborski
M.Zaborski@mini.pw.edu.pl
Michał Okulewicz
M.Okulewicz@mini.pw.edu.pl

Faculty of Mathematics and Information Science
Warsaw University of Technology

06.11.2019

Table of Contest

Research overview

Optimization problem

- **Single-objective optimization**
  - $\underset{x \in S}{\text{minimize}}\, f(x)$
- Multi-objective optimization
  - $\underset{x \in S}{\text{minimize}}\, f_i(x), i = 1, 2 \cdots n$

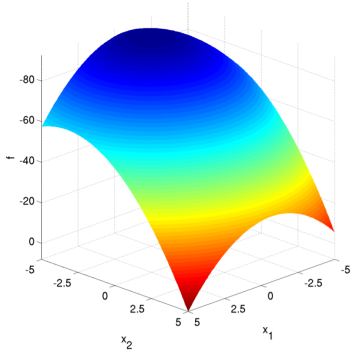


Figure 1: Sphere 2D function from COCO BBOB benchmark [1]

Optimization techniques

- Grid search
- Monte Carlo
- Simulated annealing
- Bayesian optimization
- Swarm-based algorithms (e.g. PSO)
- Evolutionary algorithms (e.g. CMA-ES)
- etc.

Optimization performance measurement

- Function evaluations
  - Especially for time-consuming evaluations
- Computation time
- Target reached
- Memory consumption

M-GAPSO

Motivation

- Particle Swarm Optimization (PSO) [2] is a well-known global optimization metaheuristic
- Nepomuceno and Engelbrecht [3] proved that appropriate mix of heterogeneous versions of PSO can lead to significant performance improvement
- Yamaguchi and Akimoto [5] presented the usage of search history for more efficient algorithm initialization after restart
- Various optimization enhancements and storing samples in memory are all promising directions of global optimization research

M-GAPSO design goals

- Rely on a well-researched optimization algorithms
- Utilize samples already gathered through randomized search procedures
- Guide the search procedure of the algorithm on the basis of already found local optima
- Maintain as much independence between those methods as possible

M-GAPSO key features

- Based on Particle Swarm Optimization (PSO)
  - Heterogeneous swarm
  - Each particle can act independently
- Adaptation mechanism (to promote best particles)
  - Can be used or not
- Restart mechanism / Initialization scheme
  - Detection of algorithm stuck
  - Changing the initial search area

M-GAPSO framework

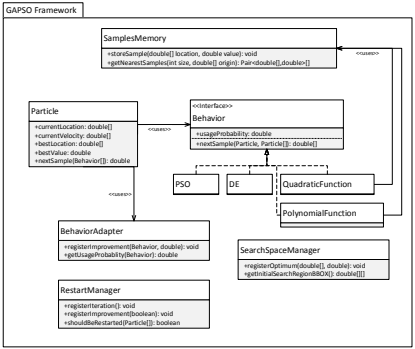


Figure 2: UML classes diagram of the proposed M-GAPSO optimization framework

M-GAPSO samples memory

- Each function evaluation is stored (as argument - value)
- Archive is limited (due to performance resons)
  - 20000 in our experiments
- When limit is reached archive is reset
- R-tree structure used in implementation

M-GAPSO heterogeneous behaviors

Current behavior pool:

- SPSO-2007
- DE/best/1/bin
- Quadratic model
  - Used during initialization
- Polynomial model

Generalized velocity update:
$particle.v_{t+1} = target.x_{t+1} - particle.x_t$

M-GAPSO adaptation

1. Observing the contributions made by each of the optimization behaviors in improvement of the global best value
2. The improvements are accumulated within a moving average
3. Normalization is applied
4. Minimum value of one particle per behavior is preserved

M-GAPSO adaptation

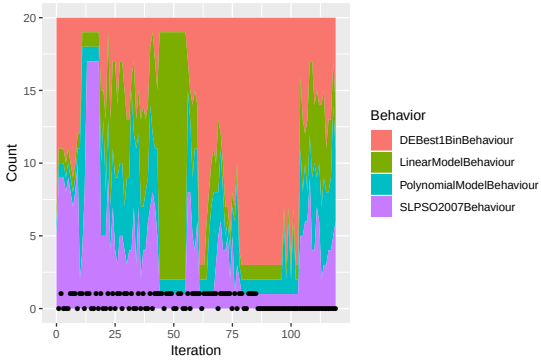


Figure 3: Behaviors utilization ratio during single algorithm run (i.e. till restart)

M-GAPSO adaptation

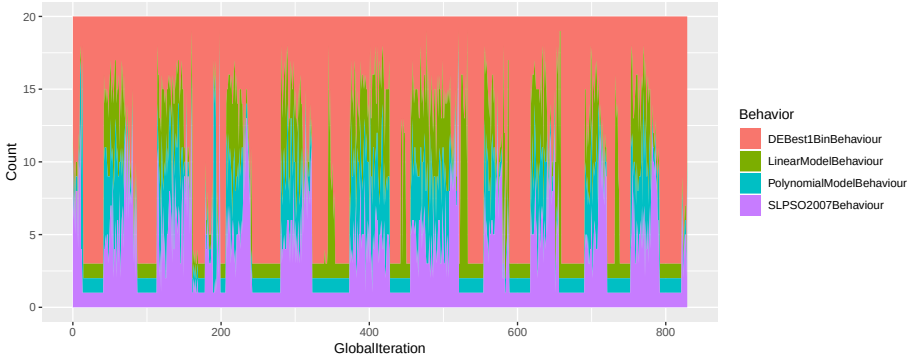


Figure 4: Overall behaviors utilization ratio for 2D multimodal function with global structure.

Model-based optimization behaviors

Quadratic and polynomial model

- Model-based behaviors (quadratic as well as polynomial models) have been applied to previous verion of algorithm - GAPSO[4]
  - To support quick convergence to local optima
- In both cases the same principles of particle's behavior are applied
  - At the beginning, the model is fitted using specified sample collection
  - Then, the algorithm finds a function optimum in relation with the field boundaries
  - Finally, the particle is moved to coordinates that match the estimated optimum

Model-based optimization behaviors

Quadratic model is fitted on a data set composed of k nearest samples (in the sense of the Euclidean metric). Quadratic function-based approach fits the following model:

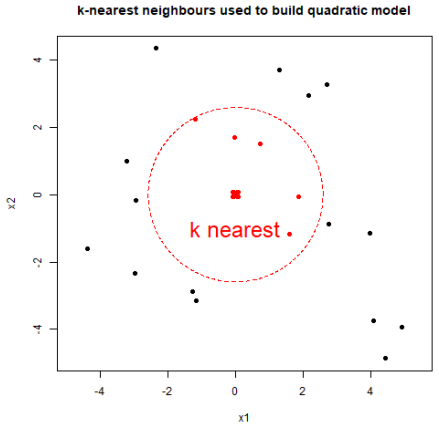
$$\hat{f}_{quadratic.local}(x) = \sum_{d=1}^{dim} \left(a_d x_d^2 + b_d x_d\right) + c \tag{1}$$

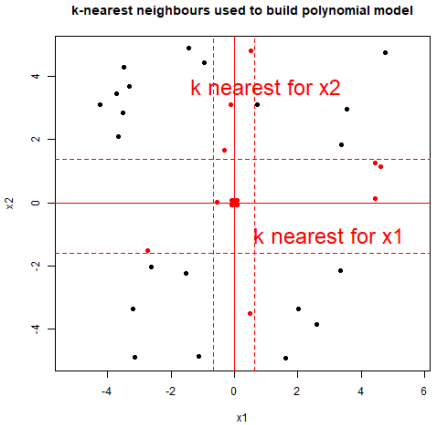
Polynomial model enhances the quadratic model in the following way:

$$\hat{f}_{polynomial.local}(x_d) = \sum_{i=1}^{p} a_{i,d} x_d^i + c \tag{2}$$

Furthermore, the polynomial model is fitted on separated data sets in each dimension.

Samples gathering





Model-based optimization behaviors

Polynomial model

- Coefficients are obtained using ordinary least squares (OLS)
  - $dims * degree$ samples
- Estimated optimum is determined on a limited range
  - According to the minimum and maximum coordinate in sample set
- Estimated optimum is determined using grid search
  - Presented results uses simple 1000 point division
  - Newest version uses Newton–Raphson algorithm starting from 100 evenly distributed points
- Numerical errors can cause problems (e.g. unstable / huge coefficients)
- Frequently polynomial is similar to linear function

Dispersion of samples

dim\dim	1	2	3	4	5
1	1.04	0.41	0.46	0.47	0.48
2	0.39	1.20	0.42	0.42	0.46
3	0.45	0.43	1.00	0.49	0.52
4	0.51	0.49	0.50	0.95	0.52
5	0.37	0.37	0.36	0.38	0.71

Figure 6: Dispersion (standard devation) of samples for f4 instance 1 for polynomial model

Dispersion of samples

Instability

Instability reasons:

- Swartm restarts
- Behaviour mixing every 10 iterations
- Particle "jumps"
- Memory resets

Possible solution:

- Detecting high volatility
- Taking more samples and detecting bad model fit

Dispersion of samples

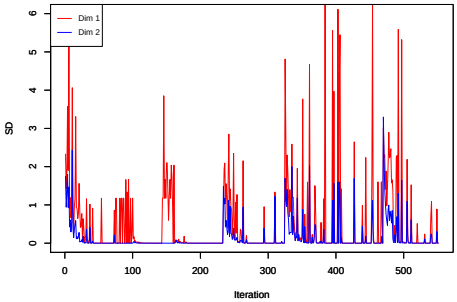


Figure 7: Dispersion of samples for f4 instance 1 for polynomial model for dimension 1

Dispersion of samples

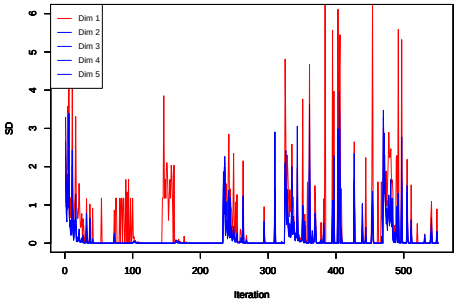


Figure 8: Dispersion of samples for f4 instance 1 for polynomial model for dimension 1

M-GAPSO performance

COCO BBOB benchmark

- Platform for optimizer comparition
- 20–100 functions (24 noiselessall)
- 5–15 instances for each function
- 2, 3, 5, 10, 20, 40 dimensions
- Up to 100 targets per instance

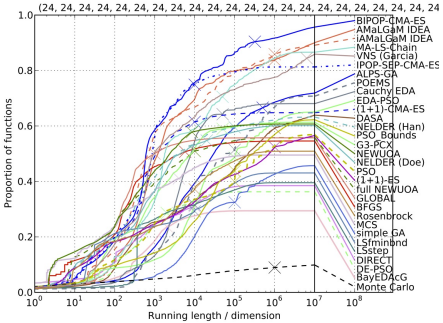


Figure 9: Example empirical runtime distributions from COCO BBOB benchmark [1]

Results

Configuration

- Probabilities:
  - PSO-L-2007: 1000
  - DE: 1000
  - LM: 1
  - PM: 1
- Population multiplier: 10
- Adaptation: false
- Samples memory: 20000
- Use cache: true
- Polynomal model degree: 4

Results

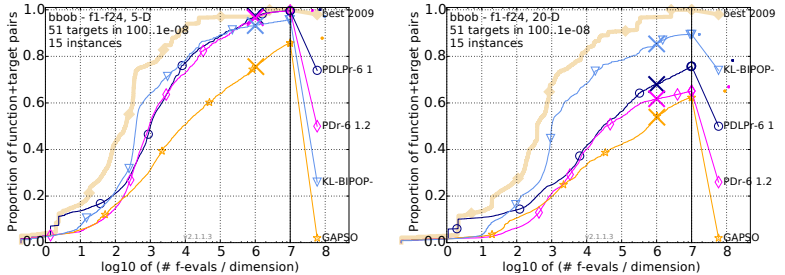


Figure 10: Results[1] of M-GAPSO configurations including model-based optimization (PDLPr) and without model based optimization (PDr) against GAPSO and state-of-the-art variation of CMA-ES.

Computation time

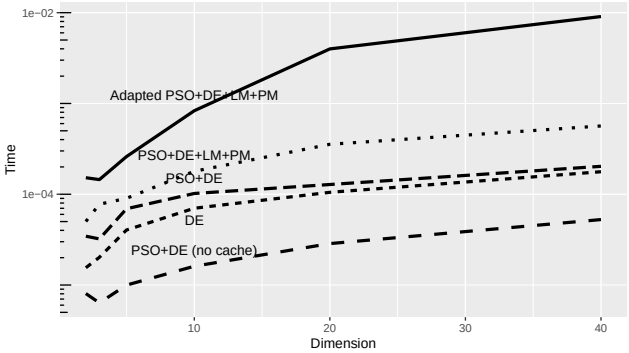


Figure 11: Average function value evaluation time of various M-GAPSO configurations with respect to function dimension.

Results

Configuration for polynomial comparition

- Probabilities:
  - PSO-L-2007: 1000
  - DE: 1000
  - LM: 1
  - PM: 0/1
- Population multiplier: 10
- Adaptation: false
- Samples memory: 20000
- Use cache: true
- Polynomal model degree: - / 2, 4, 5, 6, 10

Results for different polynomial degrees

2D and 3D

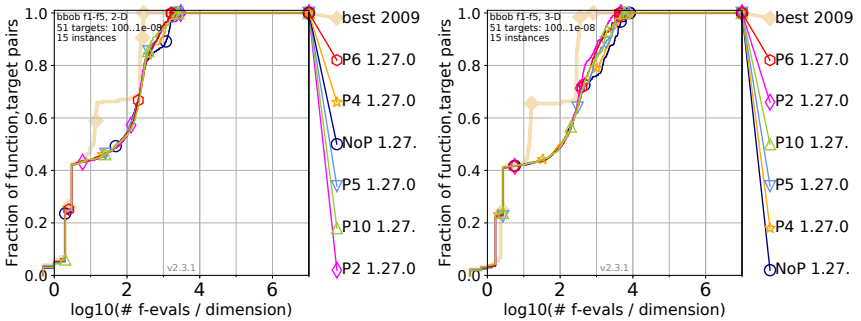


Figure 12: Results[1] of M-GAPSO configurations including model-based optimization (PDLPr) for different polynomial degrees

Results for different polynomial degrees

5D and 10D

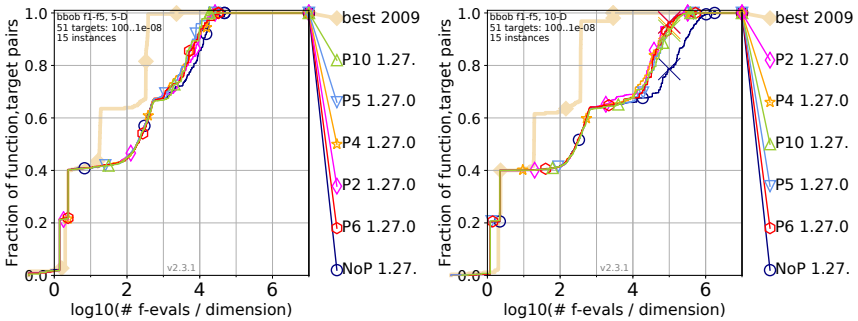


Figure 13: Results[1] of M-GAPSO configurations including model-based optimization (PDLPr) for different polynomial degrees

Future work and conclusion

Future work

- Another models can be integrated
  - Genetic algorithms
  - Bayesian optimization
  - CMA-ES
- Memory archive can be reimplemented as FIFO
- Another adaptation scheme can be implemented
- Choose between simplicity and comprehensiveness
  - Use fast PSO and DE enhancenced with efficient models
  - Use CMA-ES or other covariance algorithms

Conclusion

- The enhancements implemented in M-GAPSO improved performance over GAPSO
- Polynomial model behavior independently improved performance
- Results became comparable with the state-of-the-art CMA-ES[5],
  - Mostly for lower dimensions
- Many improvements can still be applied
  - Local models
  - Global models
  - Other mechanisms

Bibliography

- Hansen, N., Brockhoff, D., Mersmann, O., Tusar, T., Tusar, D., ElHara, O.A., Sampaio, P.R., Atamna, A., Varelas, K., Batu, U., Nguyen, D.M., Matzner, F., Auger, A.: COmparing Continuous Optimizers: numbbo/COCO on Github (2019). https://doi.org/10.5281/zenodo.2594848, https://doi.org/10.5281/zenodo.2594848
- Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Networks. IV pp. 1942–1948 (1995)
- Nepomuceno, F.V., Engelbrecht, A.P.: A Self-adaptive Heterogeneous PSO Inspired by Ants. In: International Conference on Swarm Intelligence, pp. 188–195 (2012). https://doi.org/10.1007/978-3-642-32650-9_17
- Uliński, M., Żychowski, A., Okulewicz, M., Zaborski, M., Kordulewski, H.: Generalized Self-adapting Particle Swarm Optimization Algorithm. In: Lecture Notes in Computer Science, vol. 3242, pp. 29–40. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99253-2_3
- Yamaguchi, T., Akimoto, Y.: Benchmarking the novel CMA-ES restart strategy using the search history on the BBOB noiseless testbed. In: GECCO '17 Proceedings of the Genetic and Evolutionary Computation Conference Companion. pp. 1780–1787 (2017). https://doi.org/10.1145/3067695.3084203