Asynchronous Surrogate-assisted Optimization Networks

Johannes Karder[1,2], Andreas Beham[1,2], Bernhard Werth[1,2],
Stefan Wagner[1], Michael Affenzeller[1,2]
[1]Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Hagenberg, Austria
[2]Institute for Formal Models and Verification
Johannes Kepler University, Linz, Austria
{johannes.karder,andreas.beham,bernhard.werth,stefan.wagner,michael.affenzeller}@fh-hagenberg.at

ABSTRACT

This paper introduces a new, highly asynchronous method for surrogate-assisted optimization where it is possible to concurrently create surrogate models, evaluate fitness functions and do parameter optimization for the underlying problem, effectively eliminating sequential workflows of other surrogate-assisted algorithms. Using optimization networks, each part of the optimization process is exchangeable. First experiments are done for single objective benchmark functions, namely Ackley, Griewank, Schwefel and Rastrigin, using problem sizes starting from 2D up to 10D, and other EGO implementations are used as baseline for comparison. First results show that the implemented network approach is competitive to other EGO implementations in terms of achieved solution qualities and more efficient in terms of execution times.

CCS CONCEPTS

• **Theory of computation** → **Mathematical optimization**; *Evolutionary algorithms*; • **Computing methodologies** → Supervised learning by regression;

KEYWORDS

surrogate-assisted optimization, optimization network, metaheuristic, test function, asynchronous

ACM Reference Format:
Johannes Karder, Andreas Beham, Bernhard Werth, Stefan Wagner, Michael Affenzeller. 2018. Asynchronous Surrogate-assisted Optimization Networks. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3205651.3208246

1 INTRODUCTION

In some optimization areas, especially in simulation-based optimization, objective evaluations can be quite expensive. Furthermore, depending on the problem size, conventional optimization methods often require a vast amount of evaluations to be able to find good solutions, which makes executing the required amount of (expensive) evaluations in order to explore and exploit the search space and find promising regions therein infeasible due to time constraints. Therefore, approximations of the actual objective functions, so called surrogate models, are used instead, which are orders of magnitude faster to evaluate.

In this paper, we introduce an optimization network [2] for surrogate-assisted optimization, where both the optimization of a surrogate model, as well as the optimization of problem parameters is done at the same time in an asynchronous manner. To evaluate the network approach, benchmark test functions with different problem dimensions are optimized. In order to get a baseline for comparison, the same problem instances are also optimized by two implementations of efficient global optimization (EGO) [1] in R (package: DiceOptim[1]) and HeuristicLab[2].

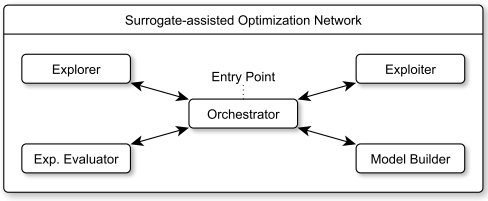

Figure 1: Conceptual view on the implemented surrogate-assisted optimization network.

2 OPTIMIZATION NETWORK

Figure 1 shows the surrogate-assisted optimization network (SAON), which consists of a total of five nodes: The *explorer* randomly explores the search space, the *exploiter*, an ALPS-GA, concentrates on particular areas of the search space with high expected improvement (EI), the *expensive evaluator* evaluates points using the underlying problem, the *model builder* creates kriging models and finally the *orchestrator* handles communication between all nodes. Once the optimization process is started, points generated by the exploration node are continuously sent to the expensive evaluator, where they are buffered in a limited FIFO queue and wait for evaluation. Each conducted expensive evaluation is returned to the orchestrator. The model builder continuously creates a surrogate model for all already known data. The surrogate model is sent

[1]https://cran.r-project.org/web/packages/DiceOptim/
[2]https://dev.heuristiclab.com/

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07…$15.00
https://doi.org/10.1145/3205651.3208246

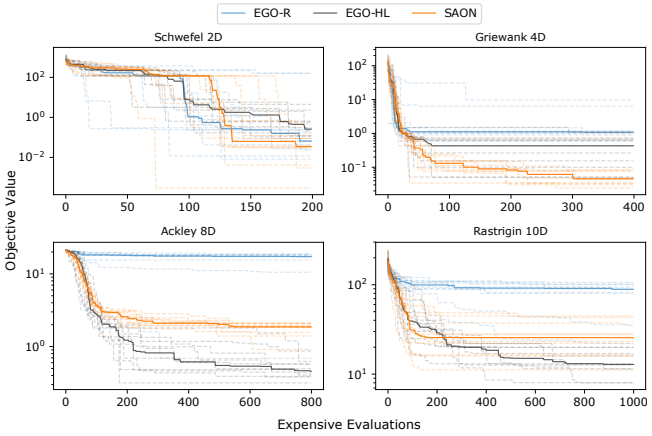

Figure 2: Some best objective value histories of EGO-R, EGO-HL and the SAON.

back to the orchestrator and, if accurate enough, used in the exploitation and expensive evaluation nodes. The points found by the exploitation node are also sent to the expensive evaluator, where they are buffered in a priority queue (priority is the EI) and wait for evaluation. All parts can be executed asynchronously.

Asynchronous optimization as shown here has its advantages and disadvantages. In synchronous optimization approaches, different steps are executed one after another in each iteration (cf. the classic EGO approach: one solution is exploited using the current model, then the model is rebuilt). If e.g. the execution time for model building increases with the number known data points, the iterations themselves will take longer which prolongs the search process. Furthermore, optimizing step by step puts parts of the algorithm to sleep until a next iteration is started. The asynchronous approach presented here keeps all parts of the network busy during the whole optimization procedure. While the explorer generates new random solutions, the exploiter focuses on certain areas of the search space, the expensive evaluator processes solutions and generates more data for the model builder, which is also repeatedly executed and continuously generates new surrogate models. In this setup, it is less hurtful if the execution time of the model builder increases, because all other parts of the network can run in the meantime. Nevertheless, this asynchronous implementation also has its drawbacks, such as race conditions that have to be taken care of and non-reproducibility of results due to thread scheduling.

3 EXPERIMENTS & RESULTS

Four well-known benchmark test functions are tested, namely the Ackley, Griewank, Rastrigin and Schwefel functions. The chosen problem sizes range from 2D up to 10D. The number of maximum expensive evaluations has been limited to $100 * D$. To simulate expensiveness, an additional evaluation delay of 5 seconds per evaluation was set. Most algorithm parameters have not been tuned. All configurations have been tested 10 times. Detailed algorithm configurations can be found on our additional material page[3].

Figure 2 shows the course of the best objective value throughout all expensive evaluations for both EGO implementations and the SAON. Dashed lines represent the actual objective values for each repetition, solid lines show the median curves. Figure 3 shows the average execution times.

[3]https://dev.heuristiclab.com/wiki/AdditionalMaterial

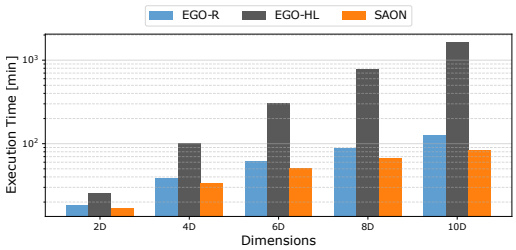

Figure 3: The average execution times of all runs in all dimensions for EGO-R, EGO-HL and the SAON.

EGO-R was faster, but inferior in terms of achieved qualities. EGO-HL consumed the most runtime, but yielded the best results in general. Quality-wise, the SAON was comparable to EGO-HL, but vastly outperformed both in terms of runtime (mind the logarithmic scale in Figure 3). The differences between the EGO variants can have a number of reasons. While EGO-R employs the usual BFGS for optimizing EI, EGO-HL uses a CMA-ES. Unlike in EGO-R, EGO-HL restarts the model building process if it resulted in a weak model. Usually, building kriging models takes longer when more data is known, but this does not seem to heavily impact EGO-R. The SAON applies parameter optimization, expensive evaluation and model building in an asynchronous manner. Therefore, e.g. optimizing EI does not block kriging, or vice versa, which positively effects execution time.

4 CONCLUSION

Benchmark test functions have been used to evaluate and compare the performance of a surrogate-assisted optimization network. The achieved results are promising and show that the network approach is competitive compared to other EGO implementations. Given the basic principle of optimization networks, it is possible to chose different algorithms for exploration, exploitation and model building. Future work includes testing the SAON with different algorithms and conduct experiments with more than one expensive evaluator in order to measure scalability.

ACKNOWLEDGMENTS

The work described in this paper was done within the *Produktion der Zukunft* Project *Integrated Methods for Robust Production Planning and Control* (SIMGENOPT2, #858642), funded by the Austrian Research Promotion Agency (FFG).

REFERENCES

[1] Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization* 13, 4 (1998), 455–492.

[2] Johannes Karder, Stefan Wagner, Andreas Beham, Michael Kommenda, and Michael Affenzeller. 2017. Towards the Design and Implementation of Optimization Networks in HeuristicLab. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '17)*. ACM, 1209–1214.