# Parameterized Analysis of Bio-inspired Computing

Frank Neumann
*Optimisation and Logistics*
*School of Computer Science*
*The University of Adelaide*
Adelaide, Australia
frank.neumann@adelaide.edu.au

***Abstract*—The parameterized analysis of bio-inspired computing provides a new way of gaining additional insights into the working behavior of popular approaches such as evolutionary algorithms and ant colony optimization. We give an overview of two important approaches in this area. The area of parameterized runtime analysis studies the runtime of bio-inspired computing with respect to different parameters of the given problem instance and builds on the success of rigorous runtime analysis of bio-inspired computing in the last 20 years. The feature-based analysis of algorithms for a given optimization problem uses statistical methods to figure out which features of a given problem instance lead to a good or bad performance of the algorithm under consideration. It often uses an evolutionary algorithm for evolving problem instances that exhibit performance differences between a given set of solvers and can be used for effective algorithm selection.**

***Index Terms*—Bio-inspired computing, evolutionary algorithms, ant colony optimization, theory, runtime analysis, features, performance**

## I. INTRODUCTION

Bioinspired computing methods such as evolutionary algorithms [7] and ant colony optimization [5] are very successful problem solvers for a wide range of combinatorial and complex engineering problems. Evolutionary algorithms follow Darwin's principle of survival of the fittest and construct solutions via an iterative process which simulates natural evolution. The basic strategy is to randomly initialize a set of possible solutions called a population and evolve it over time by creating a new population using a selection process that probabilistically favors replication of the fitter, and variation operators such as crossover and mutation. Ant colony optimization takes its inspiration from ants finding shortest paths between their nest and a food source. In the algorithmic ant colony optimization approach, solutions for a given problem are therefore constructed by artificial ants moving on a so-called construction graph. Depending on the quality of the solutions obtained, the random walk is influenced such that better solutions are constructed as the algorithm progresses.

Bio-inspired computing has found applications in various domains. However, understanding the working behaviour and analyzing its performance is a challenging task. The parameterized analysis of bio-inspired computing provides a way of analyzing bio-inspired algorithms with respect to features of a given problem instance. This allows to get insights into what type of instances a considered algorithm performs well on. Having insights into the differences in performance of different solvers on different types of problem instance is the key for effective algorithm selection.

## II. PARAMETERIZED COMPUTATIONAL COMPLEXITY ANALYSIS OF BIO-INSPIRED COMPUTING

During the last 20 years, the theoretical understanding of bio-inspired computing has tremendously advanced. The major player in the theoretical analysis of bio-inspired computing is the computational complexity analysis of evolutionary algorithms and ant colony optimization. Bio-inspired computing methods make heavy use of random decisions and this research direction treats bio-inspired computing methods as randomized algorithms. Consequently, the runtime $T$ of an algorithm is measured by the number of search points it constructs to find an optimal solution or a good approximation for a given problem. It has been shown that bio-inspired computing is provably successful for a wide range of combinatorial optimization problems as they automatically discover principles that are used in the best problem-specific algorithms. For many NP-hard problems these investigations point out worst-case instances where bio-inspired methods provably fail to obtain good results. However, instances that have to be solved in practice are usually not worst case. An important research question in the field of bio-inspired computing and combinatorial optimization is which features of a given instance determine its hardness and how doe such parameters influence the runtime of bio-inspired computing methods.

Parameterized computational complexity analysis [6] provides a mechanism for analyzing problems and algorithms by taking into account additional parameters of problem instances for a given optimization problem. This allows to provide new insights into which structural parameters make a problem easy or hard to solve. Usually, problems are formulated as decision problems in this area and an instance $(I, k)$ is given by the problem input $I$ and a parameter $k$. The question that needs to be answered is whether the answer for $(I, k)$ is "yes" or "no". Bio-inspired computing methods are designed for optimization. Therefore, one considers the time to achieve an optimal solution for a given instance $I$ in dependence of a parameter $k$ which measures the hardness of instances for a given optimization problem.

## III. FEATURE-BASED ANALYSIS OF BIO-INSPIRED COMPUTING

Feature-based analysis of combinatorial optimization problems and algorithms has become very popular in recent years. It examines in detail which properties/features of a given instance of an optimization problem make it hard or easy to solve by a given algorithm. The approach often uses an evolutionary algorithm that evolves instances that have different properties regarding the given solver(s). In many studies an evolutionary algorithm has been used to create instances where an algorithm is performing badly or very well. Furthermore, instances have been involved that exhibit a significant performance difference with respect to two given solvers. The feature values of the obtained instances are then analyzed leading to new insights into when one particular solver should be used for the optimization. Ultimately, this allows to use the feature-based insights to carry out algorithm selection.

## IV. RECENT RESULTS

We would like to highlight some results in this research area. The first parameterized computational complexity results have been obtained for bio-inspired computing and some classical NP-hard combinatorial optimization problems such as vertex cover [12], [20], makespan scheduling [25], and the Euclidean traveling salesperson problem [14], [26]. Such studies provide insights into the runtime behavior of bio-inspired computing in relation to structural parameters of the investigated problem. An important class of problems are problems that can be formulated as submodular functions. The maximization of submodular functions is NP-hard in general. This class of problems in connections with different types of Matroid constraints has been studied in [8]. The authors show that different classes of submodular functions and constraints can be approximated well by simple evolutionary algorithms and analyze the behaviour of these algorithms in dependence of different parameters of the given constraints. In the context of parameterized runtime analysis different investigations regarding the representation of solutions have been examined. Important problems that have been investigated in this context are the generalized minimum spanning tree problem [4] and the generalized traveling salesperson problem [18]. The study concerning the maximum leaf spanning tree problem [11] shows that a tree based mutation operator leads to fixed parameter evolutionary algorithms for this problem whereas standard bit-mutations don't have this property. In the context of problems with dynamically changing constraints, linear functions [21] and the vertex cover problem [17], [19] have been investigated and the runtime of evolutionary algorithms has been upper bounded taking into account the magnitude of the change of the constraints.

In the area of feature-based analysis, initial results have been obtained for local search and the traveling salesman problem. In these studies an evolutionary algorithm has been used to generate easy and hard instances for local search using the 2-opt operator [13], [24]. Furthermore, these studies have been extended in [16] by taking into account classical approximation algorithms for the Metric-TSP and carrying out a cross-comparison between the different algorithms [16]. These studies reveal different sets of features that make a TSP instance hard or easy to solve by the mentioned algorithms. In addition, they reveal complemenatry abilities of the different algorithmic approaches. The setting of different important parameters for ant colony optimization and the TSP has been studied in [15]. Furthermore, instances that discriminate between the performance of state-of-the-art solvers have been evolved [3]. One problem with evolving such instances is that the evolutionary algorithm used to create different types of instances often produces a non-diverse set of instances. Due to this, an approach has been designed which tries to fill the gaps in the feature space [23]. Furthermore, a diversity optimization approach has been introduced in [9] which allows to create instances with a given performance criteria that are diverse with respect to given features. This diversity optimization approach has also been used for the creation of similar images that vary with respect to different features [1] and feature-based diversity optimization may be of independent interest in a lot of application domains.

## ACKNOWLEDGMENT

This research has been supported by the Australian Research Council (ARC) grant DP140103400.

## REFERENCES

[1] B. Alexander, J. Kortman, and A. Neumann. Evolution of artistic image variants through feature based diversity optimisation. In Bosman [2], pages 171–178.

[2] P. A. N. Bosman, editor. *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2017, Berlin, Germany, July 15-19, 2017*. ACM, 2017.

[3] J. Bossek and H. Trautmann. Understanding characteristics of evolved instances for state-of-the-art inexact TSP solvers with maximum performance difference. In *AI*IA 2016: Advances in Artificial Intelligence - XVth International Conference of the Italian Association for Artificial Intelligence Proceedings*, pages 3–12, 2016.

[4] D. Corus, P. K. Lehre, F. Neumann, and M. Pourhassan. A parameterised complexity analysis of bi-level optimisation with evolutionary algorithms. *Evolutionary Computation*, 24(1):183–203, 2016.

[5] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, 2004.

[6] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999. 530 pp.

[7] A. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Springer, 2nd edition, 2007.

[8] T. Friedrich and F. Neumann. Maximizing submodular functions under matroid constraints by evolutionary algorithms. *Evolutionary Computation*, 23(4):543–558, 2015.

[9] W. Gao, S. Nallaperuma, and F. Neumann. Feature-based diversity optimization for problem instance classification. In Handl et al. [10], pages 869–879.

[10] J. Handl, E. Hart, P. R. Lewis, M. López-Ibáñez, G. Ochoa, and B. Paechter, editors. *Parallel Problem Solving from Nature - PPSN XIV - 14th International Conference, Edinburgh, UK, September 17-21, 2016, Proceedings*, volume 9921 of *Lecture Notes in Computer Science*. Springer, 2016.

[11] S. Kratsch, P. K. Lehre, F. Neumann, and P. S. Oliveto. Fixed parameter evolutionary algorithms and maximum leaf spanning trees: A matter of mutation. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *Parallel Problem Solving from Nature - PPSN XI, 11th International Conference, Kraków, Poland, September 11-15, 2010, Proceedings, Part I*, volume 6238 of *Lecture Notes in Computer Science*, pages 204–213. Springer, 2010.

[12] S. Kratsch and F. Neumann. Fixed-parameter evolutionary algorithms and the vertex cover problem. *Algorithmica*, 65(4):754–771, 2013.

[13] O. Mersmann, B. Bischl, H. Trautmann, M. Wagner, J. Bossek, and F. Neumann. A novel feature-based approach to characterize algorithm performance for the traveling salesperson problem. *Ann. Math. Artif. Intell.*, 69(2):151–182, 2013.

[14] S. Nallaperuma, A. M. Sutton, and F. Neumann. Parameterized complexity analysis and more effective construction methods for ACO algorithms and the euclidean traveling salesperson problem. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013*, pages 2045–2052. IEEE, 2013.

[15] S. Nallaperuma, M. Wagner, and F. Neumann. Analyzing the effects of instance features and algorithm parameters for max-min ant system and the traveling salesperson problem. *Front. Robotics and AI*, 2015, 2015.

[16] S. Nallaperuma, M. Wagner, F. Neumann, B. Bischl, O. Mersmann, and H. Trautmann. A feature-based comparison of local search and the christofides algorithm for the travelling salesperson problem. In F. Neumann and K. A. D. Jong, editors, *Foundations of Genetic Algorithms XII, FOGA '13, Adelaide, SA, Australia, January 16-20, 2013*, pages 147–160. ACM, 2013.

[17] M. Pourhassan, W. Gao, and F. Neumann. Maintaining 2-approximations for the dynamic vertex cover problem using evolutionary algorithms. In Silva and Esparcia-Alcázar [22], pages 903–910.

[18] M. Pourhassan and F. Neumann. On the impact of local search operators and variable neighbourhood search for the generalized travelling salesperson problem. In Silva and Esparcia-Alcázar [22], pages 465–472.

[19] M. Pourhassan, V. Roostapour, and F. Neumann. Improved runtime analysis of RLS and (1+1) EA for the dynamic vertex cover problem. In *IEEE Symposium Series on Computational Intelligence, SSCI 2017*, 2017. to appear.

[20] M. Pourhassan, F. Shi, and F. Neumann. Parameterized analysis of multi-objective evolutionary algorithms and the weighted vertex cover problem. In Handl et al. [10], pages 729–739.

[21] F. Shi, M. Schirneck, T. Friedrich, T. Kötzing, and F. Neumann. Reoptimization times of evolutionary algorithms on linear functions under dynamic uniform constraints. In Bosman [2], pages 1407–1414.

[22] S. Silva and A. I. Esparcia-Alcázar, editors. *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11-15, 2015*. ACM, 2015.

[23] K. Smith-Miles and S. Bowly. Generating new test instances by evolving in instance space. *Computers & OR*, 63:102–113, 2015.

[24] K. Smith-Miles, J. I. van Hemert, and X. Y. Lim. Understanding tsp difficulty by learning from evolved instances. In C. Blum and R. Battiti, editors, *LION*, volume 6073 of *Lecture Notes in Computer Science*, pages 266–280. Springer, 2010.

[25] A. M. Sutton and F. Neumann. A parameterized runtime analysis of simple evolutionary algorithms for makespan scheduling. In *Proceedings of the Twelfth Conference on Parallel Problem Solving from Nature (PPSN 2012)*, pages 52–61. Springer, 2012.

[26] A. M. Sutton, F. Neumann, and S. Nallaperuma. Parameterized runtime analyses of evolutionary algorithms for the planar Euclidean traveling salesperson problem. *Evolutionary Computation*, 22(4):595–628, 2014.