Publications of Gregor Lagodzinski

Digital Engineering · Universität Potsdam

This document lists all peer-reviewed publications of Gregor Lagodzinski, Chair for Algorithm Engineering, Hasso Plattner Institute, Potsdam, Germany. This listing was automatically generated on December 2, 2021. An up-to-date version is available online at hpi.de/friedrich/docs/publist/lagodzinski.pdf.

Journal articles

[1] Doerr, B., Kötzing, T., Lagodzinski, J. A. G., Lengler, J., The impact of lexicographic parsimony pressure for ORDER/MAJORITY on the run time. In: *Theoretical Computer Science* 816, pp. 144–168, 2020.

While many optimization problems work with a fixed number of decision variables and thus a fixed-length representation of possible solutions, genetic programming (GP) works on variable-length representations. A naturally occurring problem is that of bloat, that is, the unnecessary growth of solution lengths, which may slow down the optimization process. So far, the mathematical runtime analysis could not deal well with bloat and required explicit assumptions limiting bloat. In this paper, we provide the first mathematical runtime analysis of a GP algorithm that does not require any assumptions on the bloat. Previous performance guarantees were only proven conditionally for runs in which no strong bloat occurs. Together with improved analyses for the case with bloat restrictions our results show that such assumptions on the bloat are not necessary and that the algorithm is efficient without explicit bloat control mechanism. More specifically, we analyzed the performance of the $(1+1)$ GP on the two benchmark functions Order and Majority. When using lexicographic parsimony pressure as bloat control, we show a tight runtime estimate of $O(T_{init} + n \log n)$ iterations both for Order and Majority. For the case without bloat control, the bounds $O(T_{init} \log T_{init} + n(\log n)^3)$ and $\Omega(T_{init} + n \log n)$ (and $\Omega(T_{init} + \log T_{init})$ for $n = 1$) hold for Majority.

[2] Kötzing, T., Lagodzinski, J. A. G., Lengler, J., Melnichenko, A., Destructiveness of lexicographic parsimony pressure and alleviation by a concatenation crossover in genetic programming. In: *Theoretical Computer Science* 816, pp. 96–113, 2020.

For theoretical analyses there are two specifics distinguishing GP from many other areas of evolutionary computation: the variable size representations, in particular yielding a possible bloat (i.e. the growth of individuals with redundant parts); and also the role and the realization of crossover, which is particularly central in GP due to the tree-based representation. Whereas some theoretical work on GP has studied the effects of bloat, crossover had surprisingly little share in this work. We analyze a simple crossover operator in combination with randomized local search, where a preference for small solutions minimizes bloat (lexicographic parsimony pressure); we denote the resulting algorithm Concatenation Crossover GP. We consider three variants of the well-studied Majority test function, adding large plateaus in different ways to the fitness landscape and thus giving a test bed for analyzing the interplay of variation operators and bloat control mechanisms in a setting with local optima. We show that the Concatenation Crossover GP can efficiently optimize these test functions, while local search cannot be efficient for all three variants independent of employing bloat control.

[3] Friedrich, T., Kötzing, T., Lagodzinski, J. A. G., Neumann, F., Schirneck, M., Analysis of the (1+1) EA on Subclasses of Linear Functions under Uniform and Linear Constraints. In: *Theoretical Computer Science* 832, pp. 3–19, 2020.

Linear functions have gained great attention in the run time analysis of evolutionary computation methods. The corresponding investigations have provided many effective tools for analyzing more complex problems. So far, the runtime analysis of evolutionary algorithms has mainly focused on unconstrained problems, but problems occurring in applications frequently involve constraints. Therefore, there is a strong need to extend the methods for analyzing unconstrained problems to a setting involving constraints. In this paper, we consider the behavior of the classical (1+1) evolutionary algorithm on linear functions under linear constraint. We show tight bounds in the case where the constraint is given by the OneMax function and the objective function is given by either the OneMax or the BinVal function. For the general case we present upper and lower bounds.

Conference papers

[4] Bilò, D., Casel, K., Choudhary, K., Cohen, S., Friedrich, T., Lagodzinski, J. G., Schirneck, M., Wietheger, S., Fixed-Parameter Sensitivity Oracles. In: *Innovations in Theoretical Computer Science (ITCS)*, 2022.

We combine ideas from distance sensitivity oracles (DSOs) and fixed-parameter tractability (FPT) to design sensitivity oracles for FPT graph problems. An oracle with sensitivity f for an FPT problem Π on a graph G with parameter k preprocesses G in time $O(g(f,k)\mathsf{poly}(n))$. When queried with a set F of at most f edges of G, the oracle reports the answer to the Π-with the same parameter k-on the graph $G-F$, i.e., G deprived of F. The oracle should answer queries in a time that is significantly faster than merely running the best-known FPT algorithm on $G-F$ from scratch. We design sensitivity oracles for the k-Path and the k-Vertex Cover problem. Our first oracle for k-Path has size $O(k^{f+1})$ size and query time $O(f \min\{f, \log(f)+k\})$. We use a technique inspired by the work of Weimann and Yuster [FOCS 2010, TALG 2013] on distance sensitivity problems to reduce the space to $O\big(\big(\frac{f+k}{f}\big)^f \big(\frac{f+k}{k}\big)^k fk \cdot \log n\big)$ at the expense of increasing the query time to $O\big(\big(\frac{f+k}{f}\big)^f \big(\frac{f+k}{k}\big)^k f \min\{f,k\} \cdot \log n\big)$. Both oracles can be modified to handle vertex-failures, but we need to replace k with $2k$ in all the claimed bounds. Regarding k-Vertex Cover, we design three oracles offering different trade-offs between the size and the query time. The first oracle takes $O(3^{f+k})$ space and has $O(2^f)$ query time, the second one has a size of $O(2^{f+k^2+k})$ and a query time of $O(f+k^2)$; finally, the third one takes $O(fk+k^2)$ space and can be queried in time

$O(1.2738^k + fk^2)$. All our oracles are computable in time (at most) proportional to their size and the time needed to detect a k-path or k-vertex cover, respectively. We also provide an interesting connection between k-Vertex Cover and the fault-tolerant shortest path problem, by giving a DSO of size $O(\mathsf{poly}(f,k)\cdot n)$ with query time in $O(\mathsf{poly}(f,k))$, where k is the size of a vertex cover. Following our line of research connecting fault-tolerant FPT and shortest paths problems, we introduce parameterization to the computation of distance preservers. Given a graph with a fixed source s and parameters f,k, we study the problem of constructing polynomial-sized oracles that reports efficiently, for any target vertex v and set F of at most f edge failures, whether the distance from s to v increases at most by an additive term of k in $G-F$. The oracle size is $O(2^k k^2\cdot n)$, while the time needed to answer a query is $O(2^k f^\omega k^\omega)$, where $\omega < 2.373$ is the matrix multiplication exponent. The second problem we study is about the construction of bounded-stretch fault-tolerant preservers. We construct a subgraph with $O(2^{fk+f+k}k\cdot n)$ edges that preserves those s-v-distances that do not increase by more than k upon failure of F. This improves significantly over the $\widetilde{O}(fn^{2-\frac{1}{2^f}})$ bound in the unparameterized case by Bodwin et al. [ICALP 2017].

[5] Behrendt, L., Casel, K., Friedrich, T., Lagodzinski, J. A. G., Löser, A., Wilhelm, M., From Symmetry to Asymmetry: Generalizing TSP Approximations by Parametrization. In: *Fundamentals of Computation Theory (FCT)*. 23rd International Symposium on Fundamentals of Computation Theory (FCT 2021) (Athens, Greece,), pp. 53–66, 2021.

We generalize the tree doubling and Christofides algorithm to parameterized approximations for ATSP. The parameters we consider for the respective generalizations are upper bounded by the number of asymmetric distances, which yields algorithms to efficiently compute good approximations also for moderately asymmetric TSP instances. As generalization of the Christofides algorithm, we derive a parameterized 2.5-approximation, where the parameter is the size of a vertex cover for the subgraph induced by the asymmetric distances. Our generalization of the tree doubling algorithm gives a parameterized 3-approximation, where the parameter is the minimum number of asymmetric distances in a minimum spanning arborescence. Further, we combine these with a notion of symmetry relaxation which allows to trade approximation guarantee for runtime. Since the two parameters we consider are theoretically incomparable, we present experimental results which show that generalized tree doubling frequently outperforms generalized Christofides with respect to parameter size.

[6] Lagodzinski, J. A. G., Göbel, A., Casel, K., Friedrich, T., On Counting (Quantum-)Graph Homomorphisms in Finite Fields of Prime Order. In: *International Colloquium on Automata, Languages and Programming (ICALP)*. 48th International Colloquium on Automata, Languages, and Programming (ICALP 2021), pp. 91:1–91:15, 2021.

We study the problem of counting the number of homomorphisms from an input graph G to a fixed (quantum) graph $\bar{H}$ in any finite field of prime order $\mathbb{Z}_p$. The subproblem with graph H was introduced by Faben and Jerrum [ToC'15] and its complexity is still uncharacterised despite active research, e.g. the very recent work of Focke, Goldberg, Roth, and Zivný [SODA'21]. Our contribution is threefold. First, we introduce the study of quantum graphs to the study of modular counting homomorphisms. We show that the complexity for a quantum graph $\bar{H}$ collapses to the complexity criteria found at dimension 1: graphs. Second, in order to prove cases of intractability we establish a further reduction to the study of bipartite graphs. Lastly, we establish a dichotomy for all bipartite $(K_{3,3}\backslash\{e\}$, *domino*$)$-free graphs by a thorough structural study incorporating both local and global arguments. This result subsumes all results on bipartite graphs known for all prime moduli and extends them significantly. Even for the subproblem with $p=2$ this establishes new results.

[7] Friedrich, T., Krejca, M. S., Lagodzinski, J. A. G., Rizzo, M., Zahn, A., Memetic Genetic Algorithms for Time Series Compression by Piecewise Linear Approximation. In: *International Conference on Neural Information Processing (ICONIP)*, pp. 592–604, 2020.

Time series are sequences of data indexed by time. Such data are collected in various domains, often in massive amounts, such that storing them proves challenging. Thus, time series are commonly stored in a compressed format. An important compression approach is piecewise linear approximation (PLA), which only keeps a small set of time points and interpolates the remainder linearly. Picking a subset of time points such that the PLA minimizes the mean squared error to the original time series is a challenging task, naturally lending itself to heuristics. We propose the piecewise linear approximation genetic algorithm (PLA-GA) for compressing time series by PLA. The PLA-GA is a memetic $(\mu+\lambda)$ GA that makes use of two distinct operators tailored to time series compression. First, we add special individuals to the initial population that are derived using established PLA heuristics. Second, we propose a novel local search operator that greedily improves a compressed time series. We compare the PLA-GA empirically with existing evolutionary approaches and with a deterministic PLA algorithm, known as Bellman's algorithm, that is optimal for the restricted setting of sampling. In both cases, the PLA-GA approximates the original time series better and quicker. Further, it drastically outperforms Bellman's algorithm with increasing instance size with respect to run time until finding a solution of equal or better quality – we observe speed-up factors between 7 and 100 for instances of 90,000 to 100,000 data points.

[8] Becher, K., Lagodzinski, J. A. G., Strufe, T., Privacy-Preserving Public Verification of Ethical Cobalt Sourcing. In: *Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 998–1005, 2020.

Cobalt is a key ingredient of lithium-ion batteries and therefore is crucial for many modern devices. To ensure ethical sourcing, consumers need a way to verify provenance of their cobalt-based products, including the percentage of artisanally mined (ASM) cobalt. Existing frameworks for provenance and supply chain traceability rely on distributed ledgers. Providing public verifiability via permissionless distributed ledgers is trivial. However, offering public verifiability based on confidential production details seems contradictory. Hence, existing frameworks lack public verifiability of ratios between commodities while ensuring confidentiality of supply chain details. We propose a protocol that allows end consumers to verify the percentage of ASM cobalt in their products. Unlike previous solutions, production details are published and processed entirely in encrypted form by employing homomorphic encryption and proxy re-encryption. Thus, it ensures a high level of confidentiality of supply chain data. It has constant consumer-side complexity, making it suitable for mobile devices.

[9] Göbel, A., Lagodzinski, J. A. G., Seidel, K., Counting Homomorphisms to Trees Modulo a Prime. In: *Mathematical Foundations of Computer Science (MFCS)*, pp. 49:1–49:13, 2018.

Many important graph theoretic notions can be encoded as counting graph homomorphism problems, such as partition functions in statistical physics, in particular independent sets and colourings. In this article we study the complexity of $\#_p\mathrm{HomsTo}H$, the problem of counting graph homomorphisms from an input graph to a graph H modulo a prime number p. Dyer and Greenhill proved a dichotomy stating that the tractability of non-modular counting graph homomorphisms depends on the structure of the target graph.

Many intractable cases in non-modular counting become tractable in modular counting due to the common phenomenon of cancellation. In subsequent studies on counting modulo 2, however, the influence of the structure of H on the tractability was shown to persist, which yields similar dichotomies. Our main result states that for every tree H and every prime p the problem $\#_p\text{HomsTo}H$ is either polynomial time computable or $\#_p$P-complete. This relates to the conjecture of Faben and Jerrum stating that this dichotomy holds for every graph H when counting modulo 2. In contrast to previous results on modular counting, the tractable cases of $\#_p\text{HomsTo}H$ are essentially the same for all values of the modulo when H is a tree. To prove this result, we study the structural properties of a homomorphism. As an important interim result, our study yields a dichotomy for the problem of counting weighted independent sets in a bipartite graph modulo some prime p. These results are the first suggesting that such dichotomies hold not only for the one-bit functions of the modulo 2 case but also for the modular counting functions of all primes p.

[10] Kötzing, T., Lagodzinski, J. A. G., Lengler, J., Melnichenko, A., Destructiveness of Lexicographic Parsimony Pressure and Alleviation by a Concatenation Crossover in Genetic Programming. In: *Parallel Problem Solving From Nature (PPSN)*, pp. 42–54, 2018.

For theoretical analyses there are two specifics distinguishing GP from many other areas of evolutionary computation. First, the variable size representations, in particular yielding a possible bloat (i.e. the growth of individuals with redundant parts). Second, the role and realization of crossover, which is particularly central in GP due to the tree-based representation. Whereas some theoretical work on GP has studied the effects of bloat, crossover had a surprisingly little share in this work. We analyze a simple crossover operator in combination with local search,where a preference for small solutions minimizes bloat (lexicographic parsimony pressure); the resulting algorithm is denoted ConcatenationCrossover GP. For this purpose three variants of the well-studied Majority test function with large plateaus are considered. We show that the Concatenation Crossover GP can efficiently optimize these test functions,while local search cannot be efficient for all three variants independent of employing bloat control.

[11] Bläsius, T., Eube, J., Feldtkeller, T., Friedrich, T., Krejca, M. S., Lagodzinski, J. A. G., Rothenberger, R., Severin, J., Sommer, F., Trautmann, J., Memory-restricted Routing With Tiled Map Data. In: *Systems, Man, and Cybernetics (SMC)*, pp. 3347–3354, 2018.

Modern routing algorithms reduce query time by depending heavily on preprocessed data. The recently developed Navigation Data Standard (NDS) enforces a separation between algorithms and map data, rendering preprocessing inapplicable. Furthermore, map data is partitioned into tiles with respect to their geographic coordinates. With the limited memory found in portable devices, the number of tiles loaded becomes the major factor for run time. We study routing under these restrictions and present new algorithms as well as empirical evaluations. Our results show that, on average, the most efficient algorithm presented uses more than 20 times fewer tile loads than a normal A*.

[12] Friedrich, T., Kötzing, T., Lagodzinski, J. A. G., Neumann, F., Schirneck, M., Analysis of the (1+1) EA on Subclasses of Linear Functions under Uniform and Linear Constraints. In: *Foundations of Genetic Algorithms (FOGA)*, pp. 45–54, 2017.

Linear functions have gained a lot of attention in the area of run time analysis of evolutionary computation methods and the corresponding analyses have provided many effective tools for analyzing more complex problems. In this paper, we consider the behavior of the classical (1+1) Evolutionary Algorithm for linear functions under linear constraint. We show tight bounds in the case where both the objective function and the constraint is given by the OneMax function and present upper bounds as well as lower bounds for the general case. Furthermore, we also consider the LeadingOnes fitness function.

[13] Doerr, B., Kötzing, T., Lagodzinski, J. A. G., Lengler, J., Bounding Bloat in Genetic Programming. In: *Genetic and Evolutionary Computation Conference (GECCO)*, pp. 921–928, 2017.

While many optimization problems work with a fixed number of decision variables and thus a fixed-length representation of possible solutions, genetic programming (GP) works on variable-length representations. A naturally occurring problem is that of bloat (unnecessary growth of solutions) slowing down optimization. Theoretical analyses could so far not bound bloat and required explicit assumptions on the magnitude of bloat. In this paper we analyze bloat in mutation-based genetic programming for the two test functions ORDER and MAJORITY. We overcome previous assumptions on the magnitude of bloat and give matching or close-to-matching upper and lower bounds for the expected optimization time. In particular, we show that the $(1 + 1)$ GP takes (i) $\Theta(T_{\text{init}} + n \log n)$ iterations with bloat control on ORDER as well as MAJORITY; and (ii) $O(T_{\text{init}} \log T_{\text{init}} + n(\log n)^3)$ and $\Omega(T_{\text{init}} + n \log n)$ (and $\Omega(T_{\text{init}} \log T_{\text{init}})$ for $n = 1$) iterations without bloat control on MAJORITY.