Experimental Supplements to the Theoretical Analysis of EAs on Problems from Combinatorial Optimization

Patrick Briest, Dimo Brockhoff, Bastian Degener, Matthias Englert, Christian Gunia, Oliver Heering, Thomas Jansen, Michael Leifhelm, Kai Plociennik, Heiko Röglin, Andrea Schweer, Dirk Sudholt, Stefan Tannenbaum, and Ingo Wegener[⋆]

FB Informatik, LS2, Univ. Dortmund, 44221 Dortmund, Germany

<firstname.lastname>@uni-dortmund.de

Abstract. It is typical for the EA community that theory follows experiments. Most theoretical approaches use some model of the considered evolutionary algorithm (EA) but there is also some progress where the expected optimization time of EAs is analyzed rigorously. There are only three well-known problems of combinatorial optimization where such an approach has been performed for general input instances, namely minimum spanning trees, shortest paths, and maximum matchings. The theoretical results are asymptotic ones and several questions for realistic dimensions of the search space are open. We supplement the theoretical results by experimental ones. Many hypotheses are confirmed by rigorous statistical tests.

1 Introduction

Evolutionary algorithms (EAs) are heuristics with many applications to problems from combinatorial optimization. Most knowledge on EAs is based on intensive experiments but there are also theoretical results. Many of them consider artificial fitness functions in order to understand the search behavior of EAs. Moreover, one has to distinguish two types of theoretical results. Most of the results investigate "models of EAs" and the results have to be "verified" by experiments. Examples are the investigation of populations of infinite size, most results from schema theory, the assumption of populations in linkage equilibrium, and many more. In recent years, there is a growing interest in the rigorous analysis of the random variable describing the number of fitness evaluations until an optimal search point is evaluated. After a series of papers on artificial functions and typical fitness landscapes there are now three papers determining the asymptotic expected optimization time of mutation-based EAs on well-known problems from combinatorial optimization:

- minimum spanning trees (Neumann and Wegener (2004)),
- single source shortest paths (Scharnow, Tinnefeld, and Wegener (2002)),
- maximum matchings (Giel and Wegener (2003)).

[⋆] Supported in part by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

No such results are known for EAs with crossover.

These theoretical results give new insights into the behavior of the considered EAs and realistic problems. Nevertheless, they do not answer all questions. Asymptotic results may not describe the situation of problem dimensions considered today. Upper and lower bounds can differ and the results are concerned with the worst case and not with the typical case. For the problems listed above, we investigate properties of EAs not captured by the theoretical results and supplement the theoretical knowledge.

Section 2 introduces the investigated heuristics, mainly randomized local search (RLS) and the well-known (1+1) EA. Sections 3, 4, and 5 contain the results for the three problems and we finish with some conclusions.

The considered hypotheses have been derived from preliminary experiments which then have been tested by independent experiments. We use different statistical tests depending on the type of hypothesis under consideration. The statistical tests are carried out using the software SPSS (Version 11.5, see `www.spss.com`). Regression analysis is done using gnuplot (Version 3.7, see `www.gnuplot.info`), using the standard settings.

The statistical tests used are the Wilcoxon signed rank test (WSRT) and the Mann-Whitney test (MWT). Both are nonparametric tests to confirm the hypothesis that one random variable "systematically" produces larger values than another one. WSRT pairs the samples, takes their differences, ranks the absolute values and considers the sum of the ranks belonging to positive differences. MWT ranks all values and compares the rank sums for both samples.

Due to space limitations, we cannot present all the data (tables of data or plots). We report whether the hypotheses have been confirmed on a good significance level. More precisely, we state our results in the form "Hypothesis H has been confirmed on the significance level α" which means that the probability of producing our data if H is not true is bounded by α where α is the result of the SPSS package rounded to 3 digits. Hence, smaller values of α correspond to better results. Results are called significant, if $\alpha \leq 0.05$, very significant, if $\alpha \leq 0.01$, and highly significant, if $\alpha \leq 0.001$. Note that experiments can confirm hypotheses only for the problem dimensions n considered in the experiments and not for all n.

2 Randomized Local Search and the (1+1) EA

Most of the theoretical results for the problems listed above are for RLS and the (1+1) EA and some are for the (1+λ) EA. Therefore, we describe these algorithms. The algorithms work with "populations" of size 1, i. e., with single individuals or search points x. The first search point x is chosen uniformly at random. Then the offspring x' is obtained from x by mutation. Selection chooses x' iff $f(x') \geq f(x)$ in the case of maximization ("$\leq$" in the case of minimization). The mutation operator of RLS chooses a position of the string x uniformly at random and replaces it with a different randomly chosen value. In the case of bit strings, the bit is flipped. In the case of strings from $\{1, \ldots, k\}^n$, one of the $k-1$ different values is chosen according to the uniform distribution. The mutation operator of the (1+1) EA changes each position independently from the others with probability $1/n$. The number of chosen positions is asymptotically Poisson distributed with $\lambda = 1$ and it is a slight variant to determine first the number of positions which will be changed according to this distribution.

RLS is a hill climber with a small neighborhood and can get stuck in local optima. Indeed, RLS in its pure form is useless for spanning trees and matchings. Therefore, we have chosen for these problems a larger neighborhood of up to two local changes. RLS$_p$ changes one randomly chosen position with probability $1 - p$ and two randomly chosen positions, otherwise. We use the notation RLS for RLS$_{1/2}$.

We assume that the reader is familiar with the basic theory on the considered problems (see Atallah (1999)).

3 Minimum Spanning Trees

Minimum spanning trees can be computed efficiently by the well-known algorithms of Kruskal and Prim. Each non-optimal spanning tree can be improved by inserting a missing edge of a minimum spanning tree and excluding a more expensive edge from the cycle created by the edge insertion. Hence, there is a sequence of at most $n - 1$ "insertion plus deletion" steps which produces a minimum spanning tree with $n - 1$ edges.

These steps are local in the representation by edge sets. For graphs with m edges, the search space equals $\{0, 1\}^m$ and x describes the selection of all edges e_i where $x_i = 1$. The search space includes non-connected graphs and connected graphs with cycles. The fitness function has to penalize non-connected graphs. Let $c(x)$ be the number of connected components of the graph $G(x)$ consisting of the edge set described by x. Let $a(x)$ be the number of edges of $G(x)$ and $b(x)$ be the total weight of all chosen edges (weights are positive integers). We distinguish two fitness functions v and w for edge sets which are introduced by Neumann and Wegener (2004). Let

– $w(x) \leq w(x')$ iff $(c(x) < c(x'))$ or $(c(x) = c(x')$ and $(a(x) < a(x'))$ or $(c(x) = c(x'), a(x) = a(x')$, and $(b(x) \leq b(x'))$ and
– $v(x) \leq v(x')$ iff $(c(x) < c(x'))$ or $(c(x) = c(x')$ and $b(x) \leq b(x'))$.

The fitness functions force the search to create trees. Another possibility is to allow only trees. Prüfer numbers (see Raidl and Julstrom (2003)) are elements of $\{1, \ldots, n\}^{n-2}$ and a one-to-one coding of trees. The coding of T' obtained by an insertion and deletion of edges from T can be very different from the coding of T and it has been argued that Prüfer numbers are no useful coding. This is confirmed by experiments on graphs with $n = 10$ and random weights from $\{1, \ldots, 2i\}, 1 \leq i \leq 25$. For each i, $10\,000$ random graphs have been considered. Runs with Prüfer numbers are stopped if the number of steps is by a factor of 10 larger than the average optimization time of the (1+1) EA with edge sets and fitness function v. Runs are called successful if they find a minimum spanning tree before this time bound.

Result 1 (*MWT*)*: For $i = 1$, Prüfer numbers lead to smaller run times, for all other considered i, edge sets with fitness function v are better* (*highly significant*)*. The observed success probability is less than 0.3 for $i \geq 8$.*

For $i = 1$, there are many minimum spanning trees and Prüfer numbers have the advantage to produce only trees. It has been reported in discussions (Rothlauf (2002)) that Prüfer numbers are useful if the unique minimum spanning tree is a star, i. e., all edges adjacent to vertex 1 have weight 1, all other edges have a larger weight (random values). This has been investigated for $10 \le n \le 15$.

Result 2 (*MWT*)*: For all considered n and star-like minimum spanning trees, the average run time is smaller for edge sets and the median of the run times is smaller for Prüfer numbers* (*highly significant*).

Mean and median often yield the same kind of result. The significant differences here suggest a larger variance of run times for Prüfer numbers.

Choosing edge sets we still have the choice between the two fitness functions v and w. We have investigated the (1+1) EA on v and w for random graphs on $4i, 3 \le i \le 12$, vertices. Each edge exists with probability $1/2$ and the weights of existing edges are chosen randomly from $\{1, \ldots, n\}$. For each n, we have considered 500 random graphs. The average numbers of fitness evaluations do not differ much, they are smaller for v with the exception of $n = 24$ and $n = 44$. Only the result for $n = 32$ was confirmed as significant (MWT). Nevertheless, we conjecture that v is the better fitness function. We have chosen the value $n = 24$ (which was the worst for v in the first experiment) and have repeated the experiment with a larger number of approximately $25\,000$ random graphs.

Result 3 (*MWT*)*: For $n = 24$ and the* (*1+1*) *EA, the number of fitness evaluations is smaller for the fitness function v* (*highly significant*).

After finding some spanning tree, it is typically improved by exchanging one edge in the tree with some edge currently not in the tree. Waiting for such mutations takes some time. The fitness function v allows the algorithm to add additional edges in those steps thereby increasing their probability. Removing the additional edges from the tree can be done by mutations of single bits which are found much faster. Thus, we conjecture the same result for all other values of n but corresponding experiments are time consuming. Based on these experiments, all later experiments use the fitness function v.

Neumann and Wegener (2004) have proved a bound of $O(m^2(\log n + \log w^*))$ on the worst-case expected optimization time for graphs on m edges where weights are chosen from $\{1, \ldots, w^*\}$. We have investigated the (1+1) EA for $5 \le n \le 40$ on random graphs where each edge exists with probability $1/2$ independently from the others and weights are chosen randomly from $\{1, \ldots, w^*\}$. We have considered six values for w^* depending on the expected number of edges $\overline{m} := n(n-1)/4$: $w^* \in \{1, 2, \log \overline{m}, \overline{m}^{1/2}, \overline{m}, \overline{m}^2\}$ where $\log \overline{m}$ and $\overline{m}^{1/2}$ are rounded. The experiments are based on $1\,000$ random graphs for each w^*.

Result 4 (*MWT*)*: For all considered n, the average run time of the* (*1+1*) *EA is larger for larger values of w^** (*highly significant with two exceptions, the comparison between $\log \overline{m}$ and $\overline{m}^{1/2}$ for $n \le 15$ and the comparison between $\overline{m}$ and $\overline{m}^2$*).

The values of $\log \overline{m}$ and $\overline{m}^{1/2}$ are very close for $n \le 12$. If there are enough weights (say $\overline{m}$) such that most edges have different weights, further possible weight values have not much influence. We conjecture that the asymptotic bound can be improved to $O(m^2 \log n)$. We have investigated by regression analysis whether the data can be represented best by functions of types $c\overline{m}^{1/2}, c\overline{m}, c\overline{m}^{3/2}, c\overline{m}^2$ or $c\overline{m}^2 \log \overline{m}$. This is indeed $c\overline{m}^2 \log \overline{m}$ for $w^* \in \{\overline{m}^{1/2}, \overline{m}, \overline{m}^2\}$ and $c\overline{m}^2$ for $w^* = \log \overline{m}$. Only the other cases are much easier than the worst-case bound, namely $c\overline{m}^{3/2}$ for $w^* = 1$ and $c\overline{m}$ for $w^* = 2$.

Theory on minimum spanning trees leads to the conjecture that 2-bit flips are essential, some 1-bit flips are necessary to obtain the correct number of edges and k-bit flips, $k \geq 3$, are almost useless. The probability of a 2-bit flip is $1/2$ for RLS and approximately $1/(2e)$ for (1+1) EA. Therefore, we expect that RLS beats the (1+1) EA by a factor of nearly $e \approx 2.7$ and RLS$_{0.99}$ beats the (1+1) EA by a factor of nearly $2e \approx 5.4$. This has been tested for $1\,000$ random graphs each on n vertices, $5 \leq n \leq 36$, and $w^* = n$.

Result 5 (*MWT*)*: For all considered n, the quotient of the run times of the* (*1+1*) *EA and RLS is in* $[2.4, 2.8]$ (*significant for $n \geq 15$ and $n \neq 30$*) *and for the* (*1+1*) *EA and RLS*$_{0.99}$ *in* $[5.2, 5.9]$ (*significant for $n \geq 22$ and $n \neq 36$*)*.*

There seem to be some special effects for small n and we have to expect some negative test results among many tests. In any case, it is highly significant (MWT) that RLS$_{0.99}$ is faster than RLS which is faster than the (1+1) EA.

Neumann and Wegener (2004) have proved that the expected number of fitness evaluations for the worst-case instance is asymptotically the same for the (1+λ) EA as long as $\lambda \leq m^2/n$. The (1+λ) EA is faster on parallel computers. Our experiments confirm that larger values of λ increase the number of fitness evaluations but the effect tends to vanish for larger n.

Result 6 (*MWT*)*: For all considered n, the average number of fitness evaluations of the* (*1+1*) *EA is smaller than for the* (*1+10*) *EA but these differences are not significant. For $n \leq 20$, the* (*1+1*) *EA is faster than the* (*1+100*) *EA (highly significant) and for larger n, the effect gets less significant.*

An always essential question is whether a theoretical worst-case analysis captures the typical case. Neumann and Wegener (2004) present a family of graphs (with only three different weight values) where RLS and the (1+1) EA need an expected number of $\Theta(m^2 \log n)$ fitness evaluations. We have investigated the worst-case graphs on $n = 4i + 1$ vertices, $1 \leq i \leq 10$, which have $2i^2 + 4i$ edges (240 edges for the largest graph). Then we have compared $1\,000$ runs of the (1+1) EA for each n with $1\,000$ runs on random graphs with the same values of n and m and random weights from $\{1, \ldots, 1\,000\}$.

Result 7 (*MWT*)*: For all considered n, the* (*1+1*) *EA is faster on the "asymptotic worst-case instance" than on random graphs* (*highly significant*)*.*

A regression analysis proves that $cm^2 \log n$ describes the data for random graphs better than $cm, cm \log m, cm^2$ or cm^3 as is shown in Figure 1. The observable differences are in accordance with differences in mean square errors. This holds for other places where we present the results of regression analyses, too. Here, we arrive at the interesting conjecture that $\Theta(m^2 \log n)$ is the typical run time of the (1+1) EA for the computation of minimum spanning trees.

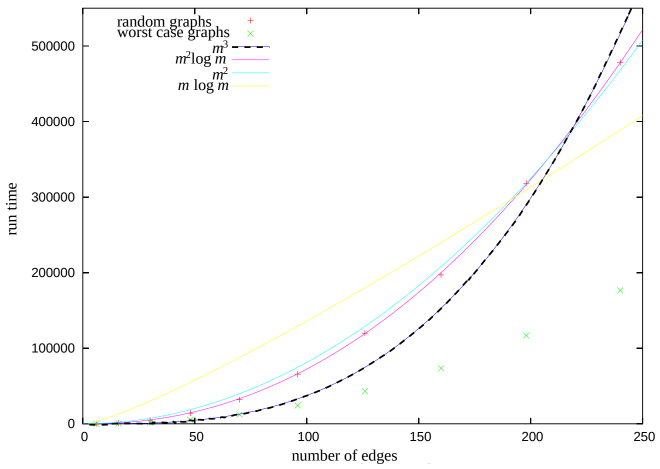

Fig. 1. Regression analysis of average run times of the (1+1) EA for the MST on random graphs.

4 Single Source Shortest Paths

The problem is to compute for a complete graph and a distinguished vertex s shortest paths from s to all other vertices. This problem is solved by Dijkstra's algorithm. Scharnow, Tinnefeld, and Wegener (2002) have investigated the search space $\{1, \dots, n\}^{n-1}$ where x_i is the predecessor of vertex i and $s = n$. Dijkstra's algorithm computes solutions which can be described by trees rooted at n and have such a representation. The search space contains also graphs with cycles. There are two fitness functions, a single-criterion function measuring the sum of the distances of all n-i-paths and a multi-criteria function with the vector of all n-i-path lengths. Then we are interested in the unique Pareto optimal fitness vector and "$\leq$" means "$\leq$" for all vector positions. Scharnow et al. have shown that the single-criterion case contains instances with a behavior like the needle in a haystack if non-existing paths are penalized by ∞. They proved an $O(n^3)$ bound for the multi-criteria case which is tight for worst-case instances. For graphs where shortest paths do not contain too many edges, one can expect an $O(n^2 \log n)$ bound. Here, we penalize a non-existing path by a distance which is larger than the length of a longest path.

The multi-criteria fitness function contains more information but $f(x') \geq f(x)$ holds only if no n-i-path for x' is longer than for x. The question is which fitness function supports better the optimization by the (1+1) EA. We have investigated graphs on $n = 10i$ vertices, $1 \leq i \leq 15$, where the weights are chosen randomly from $\{1, \dots, k\}, k = 2^j, 1 \leq j \leq \lfloor 2 \log n \rfloor + 1$.

Result 8 (*MWT*)*: For all considered n and k, the* (*1+1*) *EA is faster with the single-criterion fitness function* (*highly significant*).

For both fitness functions, a regression analysis for functions $an^3 + bn^2 \log n + cn^2$ leads to a good fit with a-values very close to 0.

For the shortest paths problem as well as for the problem of computing minimum spanning trees, we have compared how our heuristics work on two different fitness functions. In both cases, the algorithms work better for the fitness function containing less information. This seems to be a remarkable result since it is in contrast to the plausible idea that the fitness function should contain as much information as possible.

5 Maximum Matchings

Given a graph, an edge set is a matching iff no two edges share a vertex. We use a representation by edge sets and non-matchings are penalized according to the number of conflicts, i. e., the number of edge pairs which share a vertex. Otherwise, the fitness describes the size of the matching. Matchings cannot be improved always locally but along so-called augmenting paths leading from a free vertex (not matched by the chosen edge set) to another free vertex where non-chosen and chosen edges alternate (see Figure 2). Giel and Wegener (2003) have proved that RLS and the (1+1) EA are

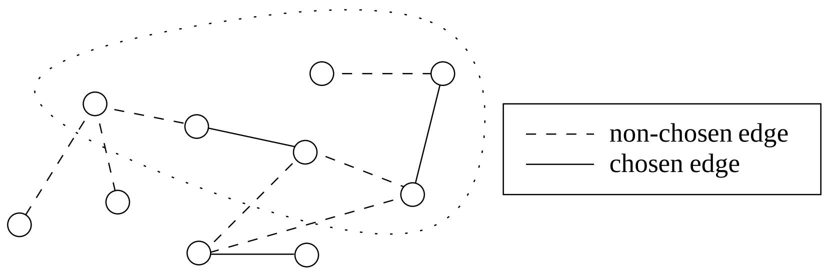

Fig. 2. An example of a graph with an augmenting path of length 5.

polynomial-time randomized approximation schemes, i. e., for each constant $\varepsilon > 0$, a matching M whose size is at least $(1 - \varepsilon) \cdot |M_{\text{opt}}|$ for a maximum matching M_{opt} is obtained in expected polynomial time. However, the degree of the polynomial depends on $1/\varepsilon$. There are worst-case examples where both algorithms need on average exponential time to compute an optimal matching. Moreover, they have analyzed the expected optimization time on simple graphs. Graphs which are consisting of one path of length n are of special interest since, despite of their simpleness, one may expect to obtain augmenting paths of linear length. Based on results for the gambler's ruin problem (Feller (1968)), Giel and Wegener (2003) have proved an $O(n^4)$-bound for the expected optimization time of RLS and the (1+1) EA. This bound is tight if we obtain a search point containing an augmenting path of linear length.

As in the case of minimum spanning trees, 2-bit flips seem to be the essential steps to shorten augmenting paths and 1-bit flips can improve the matching if a free edge (an edge between two free vertices) exists. Hence, we expect that RLS is better than the (1+1) EA. We have investigated paths on $10i$ vertices, $1 \leq i \leq 10$, and have performed $1\,000$ runs for each n.

Result 9 (*MWT*)*: For all considered n, RLS is faster than the* (*1+1*) *EA on a path* (*highly significant*)*.*

Again, we may expect a gain by a factor of approximately e. The average factor falls into $[2.3, 3.1]$ for $n \geq 30$ but the variance is too large to obtain significant results.

A regression analysis with functions $an^4 + bn^3 + cn^2 + dn$ shows that the n^4-term is necessary for a good fit. However, the n^4-term has a quite small value (as also the n^3-term). This supports the observation that there are long (linear length) augmenting paths but they are essentially shorter than n. Paths are so simple that one may conjecture that a GA with diversity-preserving methods may beat the (1+1) EA. This may be the case for two-point crossover which can replace a subpath from one individual by the corresponding subpath of another individual. However, this crossover operator is specialized to paths and not appropriate for more general graphs. Since we are not interested in this paper in problem-specific search operators, we do not discuss experiments with crossover.

Paths are special trees. Giel and Wegener (2003) have shown an $O(n^6)$ bound on the expected optimization time of RLS for finding a maximum matching on trees with n vertices but conjecture an $O(n^4)$ bound for RLS and the (1+1) EA. Trees may have a smaller diameter than paths with the same number of vertices and the size of the maximum matching typically is smaller than for paths. However, there are situations where, for trees, it is more likely to lengthen augmenting paths than to shorten them (see Figure 3). For paths, this cannot happen. Random trees can be generated by choosing

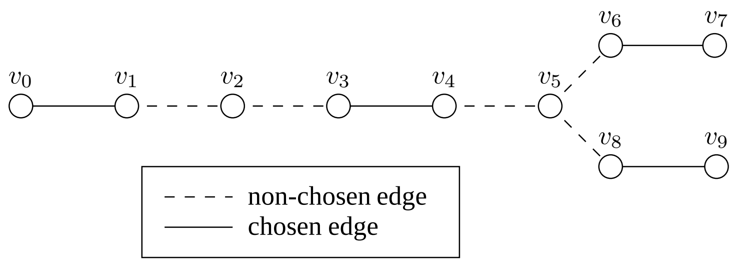

Fig. 3. An augmenting path $p = v_2, \ldots, v_5$ where RLS has at v_2 one possibility to lengthen p and one possibility to shorten p and at v_5 one possibility to shorten p and two possibilities to lengthen p.

randomly a Prüfer number. For $n = 10i, 1 \leq i \leq 10$, we have compared $1\,000$ runs of the (1+1) EA on the path with the (1+1) EA on $1\,000$ random trees.

Result 10 (*MWT*)*: For all considered n, the* (*1+1*) *EA is faster on random trees than on paths* (*highly significant*)*.*

It can be even confirmed that the (1+1) EA is faster on random trees on 200 vertices than on paths with 100 vertices. Regression analysis shows that the data for random trees can be well expressed by a polynomial of degree 3 while the data for paths needs an n^4-term.

Giel and Wegener (2002) have proved an $O(n^{3.5})$ bound for k-ary trees which are rooted trees of minimal depth if the number of children is k (at most k on the last two levels). Since the depth decreases with k, also the diameter and the maximal length of augmenting paths decrease with k which should lead to decreased run times. We have investigated random trees and k-ary trees on $10i$ vertices, $1 \leq i \leq 20$. The data confirms the following list of increasing difficulties for the (1+1) EA: 5-ary, 4-ary, ternary, binary, random. However, many comparisons were not significant. Indeed the run times for k-ary trees do not increase with n. There are significant decreases. All results depend on the positions of the leaves on the last level (see Figure 4). Assigning them from left to right leads to much smaller maximum matchings than assigning them randomly. Smaller maximum matchings can be found faster. More vertices concentrated in a small

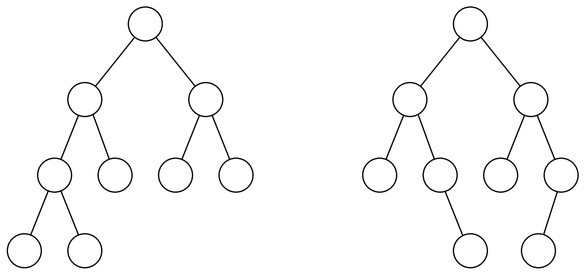

Fig. 4. Examples for the two strategies to assign the position of the leaves on the last level.

subtree may not increase the size of a maximum matching but may increase the number of maximum matchings which can simplify the optimization task. This has to be examined further and is beyond the scope of this paper.

Finally, we have followed a quite new research direction in theoretical computer science, namely the investigation of semirandom problem instances. The idea is to investigate problem instances of the following kind. First, an optimal solution is planted into the problem instance and then, randomly, further objects are added to the problem instance, see Feige and Kilian (2000) for a survey on these problem types and Feige and Krauthgamer (2000) for the problem of finding a large hidden clique in a semirandom graph.

In our case, we fix the number n of vertices and the number m of edges. Then we deterministically choose the edges of a perfect matching and, afterwards, we choose randomly $m - \lfloor n/2 \rfloor$ further edges. Since the dimension of the search space is m, we compare different values of n for fixed m. If n is large, the "planted" perfect matching typically is the only maximum one and the few further edges do not prevent an efficient search. If n is small, the many further edges produce many perfect matchings and it is easy to find one of them. Hence, it is expected, that the expected run times of the (1+1) EA are first increasing with n and decreasing after its maximal value. The experiments have considered the cases $m = 50i, 1 \leq i \leq 4$, and 100 runs for the different values of n. The differences between neighbored values of n are too small to be determined as very significant. It is convenient to compare values n and n' of small difference.

Result 11 (*WSRT*)*: For the considered m, the average run time of the* (*1+1*) *EA on semirandom graphs is increasing with n until a maximum value is reached and then decreasing* (*significant, many subresults are very or even highly significant, the results are not significant close to the maximum value*).

Close to the maximum value, the gradient of the function has small absolute value and it is not surprising to obtain ambiguous results in this range. These experiments should lead to further theoretical studies in order to obtain asymptotic values for various parameter constellations described by functions between m and n, e. g., $n = m^{\alpha}$, $1/2 < \alpha < 1$.

Conclusions

The theoretical analysis of simple heuristics on some of the best-known problems from combinatorial optimization has given a coarse picture which has been refined by our experiments. For minimum spanning trees and shortest paths, we have decided (for small dimensions of the search space) which fitness function is better suited. For many cases, we have decided which heuristic is better. In some cases, we have classified problem instances according to their difficulty for the considered heuristics. For minimum spanning trees, our experiments lead to the conjecture that worst case and typical case are asymptotically of equal difficulty. Finally, we have started experiments on semirandom problem instances, a promising research direction for experiments and theory.

References

1. Atallah, M. J. (Ed.) (1999). *Algorithms and Theory of Computation Handbook.* CRC Press.
2. Feige, U. and Kilian, J. (2000). Heuristics for semirandom graph problems. Journal of Computer and System Sciences 63, 639–671.
3. Feige, U. and Krauthgamer, R. (2000). Finding and certifying a large hidden clique in a semirandom graph. Random Structures and Algorithms 16, 195–208.
4. Feller, W. (1968). *An Introduction to Probability Theory and Its Applications.* John Wiley & Sons.
5. Giel, O. and Wegener, I. (2003). Evolutionary algorithms and the maximum matching problem. Proc. of 20th Symp. on Theoretical Aspects of Computer Science (STACS), LNCS 2607, 415–426.
6. Neumann, F. and Wegener, I. (2004). Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. Proc. of Genetic and Evolutionary Computation Conference (GECCO 2004), LNCS 3102, 713–724.
7. Raidl, G. R. and Julstrom, B. A. (2003). Edge sets: an effective evolutionary coding of spanning trees. IEEE Trans. on Evolutionary Computation 7, 225–239.
8. Rothlauf, F. (2002). *Representations for Genetic and Evolutionary Algorithms.* Physica-Verlag.
9. Scharnow, J., Tinnefeld, K., and Wegener, I. (2002). Fitness landscapes based on sorting and shortest paths problems. Proc. of the 7th Conf. on Parallel Problem Solving from Nature (PPSN-VII), LNCS 2439, 54–63.