UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

Fitness Landscapes Based on Sorting and Shortest Path Problems

Jens Scharnow, Karsten Tinnefeld, and Ingo Wegener

No. CI-128/02

Technical Report ISSN 1433-3325 March 2002

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI 44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence", at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

Fitness Landscapes Based on Sorting and Shortest Paths Problems

Jens Scharnow, Karsten Tinnefeld*, and Ingo Wegener*

FB Informatik, LS2, Univ. Dortmund, 44221 Dortmund, Germany
wegener@ls2.cs.uni-dortmund.de

Abstract. The analysis of evolutionary algorithms is up to now limited to special classes of functions and fitness landscapes. It is not possible to describe those subproblems of NP-hard optimization problems where certain evolutionary algorithms work in polynomial time. Therefore, fitness landscapes based on important computer science problems as sorting and shortest paths problems are investigated here. Although it cannot be expected that evolutionary algorithms outperform the well-known problem specific algorithms on these simple problems, it is interesting to analyze how evolutionary algorithms work on these fitness landscapes which are based on practical problems. The following results are obtained:

- Sorting is the maximization of "sortedness" which is measured by one of several well-known measures of presortedness. The different measures of presortedness lead to fitness landscapes of quite different difficulty for EAs.
- Shortest paths problems are hard for all types of EA, if they are considered as single-objective optimization problems, while they are easy as multi-objective optimization problems.

1 Introduction

Our aim is to contribute to a theory of evolutionary algorithms (EAs) which analyzes the expected optimization time of EAs on important and interesting problems and classes of fitness functions. Nowadays, it is a vision to explain the success of EAs on hard problems by identifying those instances of the problem where the considered EA finds the optimum in expected polynomial time. In order to develop tools for such results EAs have to be analyzed in various situations.

Fitness landscapes considered as typical ones have been described and analyzed in many papers (for an overview see Bäck, Fogel, and Michalewicz (1997)). Moreover, interesting classes of fitness functions have been investigated, e.g., separable functions (Droste, Jansen, and Wegener (1998a)), monotone polynomials of small degree (Wegener (2001)), long-path functions (Horn, Deb, and Goldberg (1994), Rudolph (1997), Droste, Jansen, and Wegener (1998b)), and

* This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

royal road functions (Mitchell, Holland, and Forrest (1994), Jansen and Wegener (2001a)). However, these are artificial functions and problems.

Here we choose the approach to investigate EAs on the most basic and important computer science problems, namely sorting (the maximization of the sortedness) and shortest path problems. We do not and cannot expect EAs to outperform Quicksort or Dijkstra's algorithm. However, universal problem solvers like EAs should be efficient on these simple problems.

Sorting is the problem of maximizing the sortedness. Hence, the fitness of a permutation can be measured by one of the well-known measures of presortedness. In Section 2, the corresponding fitness landscapes are introduced and appropriate mutation operators are discussed. The analysis in Section 3 shows that most measures of presortedness contain enough information to direct the optimization by EAs. However, there is a well-known measure of presortedness leading to a fitness landscape with large plateaus of search points having equal fitness such that the optimization process gets stuck on such a plateau.

Shortest path problems are multimodal optimization problems and EAs get stuck in local but not global optima (for certain instances). In Section 4, we describe this fitness landscape and an alternative as multi-objective optimization problem. Moreover, we prove that only the multi-objective optimization problem description directs the search of EAs efficiently. This is the first result of this type for EAs.

2 Fitness Landscapes Based on Sorting Problems

Given a sequence of n distinct elements from a totally ordered set, sorting is the problem of maximizing the sortedness. Hence, the search space is the set of all permutations π on $\{1, \ldots, n\}$. For our investigations we can identify π with the sequence $(\pi(1), \ldots, \pi(n))$ and the identity permutation is the optimal one. If we try to solve sorting as such an optimization problem, we need a fitness function f such that $f(\pi)$ describes the sortedness of π. Such measures have been introduced in the discussion of so-called adaptive sorting algorithms (see, e.g., Moffat and Petersson (1995)). We will investigate the five best-known measures of presortedness and the corresponding fitness landscapes.

- $\mathrm{INV}(\pi)$ measures the number of pairs (i, j), $1 \leq i < j \leq n$, such that $\pi(i) < \pi(j)$ (pairs in correct order),
- $\mathrm{HAM}(\pi)$ measures the number of indices i such that $\pi(i) = i$ (elements at the correct position),
- $\mathrm{RUN}(\pi)$ measures the number of indices i such that $\pi(i) > \pi(i+1)$ (number of maximal sorted blocks minus 1) leading to a minimization problem,
- $\mathrm{REM}(\pi)$ equals the largest k such that $\pi(i_1) < \cdots < \pi(i_k)$ for some $i_1 < \cdots < i_k$ (length of the longest sorted subsequence),
- $\mathrm{EXC}(\pi)$ equals the minimal number of exchanges (of pairs $\pi(i)$ and $\pi(j)$) to sort the sequences again leading to a minimization problem.

Since we cannot see any advantage of crossover for sorting problems, we have investigated only mutation-based EAs. Our mutation operator is based on the following two simple operations:

- exchange(i, j) which exchanges the elements at the positions i and j,
- jump(i, j) where the element at position i jumps to position j while the other elements between position i and position j are shifted in the appropriate direction, e.g., jump(5,2) applied to (6,4,3,1,7,2,5) produces (6,7,4,3,1,2,5).

Mutation should allow any π' to be produced from π with positive probability. The usual mutation operator on the search space $\{0,1\}^n$ flips each bit with probability $1/n$. This implies that the number of flipping bits is asymptotically Poisson distributed with parameter $\lambda = 1$. Therefore, we have chosen the following mutation operator where we exclude steps where nothing happens:

- Choose s according to a Poisson distribution with parameter $\lambda = 1$ and perform sequentially $s+1$ exchange or jump steps where for each step (i, j) is chosen randomly among all pairs (k, l), $1 \leq k, l \leq n$, $k \neq l$, and it is decided randomly whether exchange(k, l) or jump(k, l) is performed.

We also may consider only exchange or only jump steps. Finally, we have decided to analyze the following evolutionary algorithm shortly called $(1 + 1)$EA which resembles the well-known (1+1) evolution strategy:

- Choose the first search point π randomly.
- Repeat: Produce π' by mutation from π and replace π by π' if π' is not worse than π ($f(\pi') \geq f(\pi)$ in the case of a maximization problem and $f(\pi') \leq f(\pi)$ otherwise).

In applications, one needs a stopping criterion. Here we consider the infinite stochastic process described above and investigate the random variable called optimization time which equals the first point of time when π is optimal.

3 The Analysis of the (1+1)EA on the Fitness Landscapes Based on Sorting Problems

Here we analyze the performance of the (1+1)EA on the different landscapes described by the fitness functions introduced in Section 2.

Theorem 1. *The expected optimization time of the (1+1)EA on the fitness landscape described by INV is bounded above by* $O(n^2 \log n)$ *and bounded below by* $\Omega(n^2)$.

Proof. Let π be the current search point, $1 \leq i < j \leq n$, and $\pi(i) > \pi(j)$, i.e., (i, j) is an incorrect pair. Then exchange(i, j) improves the fitness. The reason is the following. The pair (i, j) is afterwards a correct one. Let $i < k < j$. If $\pi(k) > \pi(i)$ or $\pi(k) < \pi(j)$, the correctness status of the pairs (i, k) and (k, j) is not changed. If $\pi(j) < \pi(k) < \pi(i)$, also the incorrect pairs (i, k) and (k, j)

are corrected by the exchange operation. Each specific exchange operation has a probability of $1/(2en(n-1))$ to be the only operation ($1/e$ is the probability of performing a single operation, $1/2$ the probability of choosing an exchange operation and $1/(n(n-1))$ the probability of choosing the pair (i,j)). Hence, the probability of increasing the fitness in the presence of m incorrect pairs is at least $m/(2en(n-1))$ leading to an expected waiting time (for an improvement) of $O(n^2/m)$. Since $1 \leq m \leq n(n-1)/2$ and since the fitness is never decreased, the expected optimization time can be estimated by

$$c \sum_{1 \leq m \leq n(n-1)/2} n^2/m = cn^2 H(n(n-1)/2)$$

for the harmonic series H where $H(N) \leq \ln N + 1$ leading to the proposed bound $O(n^2 \log n)$.

For the lower bound we only consider the final step leading to the optimum. Independent of the number of exchange or jump operations chosen in this step, it is necessary that the last operation changes a not sorted sequence into a sorted one. This is possible by at most one exchange and at most two jump operations ($\text{jump}(i, i+1)$ and $\text{jump}(i+1, i)$ have the same effect). Hence, the success probability of the final step is bounded above by $3/(n(n-1))$ and, therefore, the expected waiting time is bounded below by $\Omega(n^2)$. □

In the beginning the fitness typically is increased in successful steps by more than one. This may change in the final stage of the optimization process. The sequence $(2, 3, \ldots, n, 1)$ has $n-1$ incorrect pairs and, in successful steps, the number of incorrect pairs is halved on the average. This leads to an expected optimization time of $O(n^2)$. The sequence $(2, 1, 4, 3, \ldots, n, n-1)$ has $n/2$ incorrect pairs and, in successful steps, the number of incorrect pairs is decreased only by one with large probability. It can be shown that the expected optimization time on this string equals $\Theta(n^2 \log n)$. It is more typical to have in the final stage more small disordered subblocks than to have only few elements which belong to many incorrect pairs. Hence, we believe that the expected optimization time equals $\Theta(n^2 \log n)$.

Corollary 1. *The expected optimization time of the $(1+1)EA$ is $\Omega(n^2)$ for each fitness landscape based on the sorting problem, i.e., with a unique global optimum.*

Proof. The proof is contained in the proof of Theorem 1, since there we only used the fact that there is a unique global optimum. □

Theorem 2. *The expected optimization time of the $(1+1)EA$ on the fitness landscape described by REM equals $\Theta(n^2 \log n)$.*

Proof. If $\text{REM}(\pi) = k$ and j is a position not belonging to the longest sorted subsequence, then there is a jump operator where j jumps to a position where it increases the length of the sorted subsequence from k to $k+1$. The number of

these positions j equals $n-k$ and, therefore, the success probability (increasing the fitness), if $\mathrm{REM}(\pi) = k$, is at least $(n-k)/2en(n-1)$ leading to an expected waiting time of $O(n^2/(n-k))$. Summing up these values for $k \in \{1, \ldots, n-1\}$ we obtain the proposed bound.

For the lower bound it is essential that a single jump can change the REM value by at most 1. The jumping element is first taken from the sequence (which can decrease the length of any sorted subsequence by 1) and then inserted somewhere (which can increase the length of sorted subsequences by 1). Since an exchange can be simulated by two jumps, exchange steps can increase the REM value by at most 2. Let us consider the situation where $\mathrm{REM}(\pi) \geq n - n^{1/2}$ for the first time. Then $\mathrm{REM}(\pi) \leq n - n^{1/2}/2$ with overwhelming probability, since at least $n^{1/2}/4$ jumps and exchanges in one step are very unlikely. If an element of the longest sorted subsequence which has two neighbors from this sorted subsequence jumps, this decreases the REM value by 1 while the probability that an element outside the sorted subsequence jumps to a position increasing the REM value equals $1/(n-1)$ for each of these values. We can conclude that the probability of increasing the REM value by a step with at least 10 exchanges and/or jumps is so small that this does not happen within $O(n^2 \log n)$ steps with overwhelmig probability. This implies that we may consider only steps with a single jump operation (increasing the expected optimization time at most by a constant factor). If $\mathrm{REM}(\pi) = k$, the success probability is bounded above by $(n-k)/(n(n-1))$ leading to a waiting time of $\Omega(n^2/(n-k))$. Summing up for $k = n - n^{1/2}/2$ to $k = n-1$, we obtain the proposed lower bound. □

The upper bound of Theorem 2 holds for an arbitrary initial search point π. However, jump operations are essential while the bounds of Theorem 1 for INV even hold if we perform only jumps or only exchanges. Consider the situation of $\pi^* = (2, \ldots, n, 1)$ with $\mathrm{REM}(\pi^*) = n-1$. The plateau of search points π with $\mathrm{REM}(\pi) = n-1$ can be characterized by the "wrong element" i which can be at one of the $n-1$ "wrong positions" $p \neq i$ while all other elements are in sorted order. Which exchange operations are accepted? Only those which exchange i with one of the two neighbored elements and those which exchange i with one of the elements $i-1$ or $i+1$. Hence, the probability of an accepted exchange step is bounded above by $4/(n(n-1))$. The probability of accepting a step with more than one exchange step is negligible. However, the sum of all $|j - \pi(j)|$ is decreased at most by 2 by a successful exchange step and we need $(n-1)/2$ of these steps for π^* leading already to a waiting time of $\Omega(n^3)$. However, using the methods introduced by Jansen and Wegener (2001b) we can prove that the expected optimization time equals $\Theta(n^4)$ showing that the choice of the mutation operator is essential.

Theorem 3. *The expected optimization time of the* $(1+1)$*EA on the fitness landscape described by HAM is bounded above by* $O(n^2 \log n)$.

Proof. The upper bound follows in the same way as in the proof of Theorem 2 by proving that for $\mathrm{HAM}(\pi) = k$ there are at least $n-k$ single exchange operations increasing the HAM value. If element i is at position $p \neq i$, then exchange(i, p)

increases the HAM value. Element i reaches its correct position and the element leaving position i was not at its correct position. □

Considering only exchanges we can also prove a lower bound of $\Omega(n^2 \log n)$ similarly to the lower bound proof of Theorem 2. Here the HAM value can be increased by a single exchange step at most by 2. If $\mathrm{HAM}(\pi)$ is large, it is unlikely that a step with more than 10 exchanges increases the HAM value. Moreover, only exchanges of pairs (i, j) where position j contains i and/or position i contains j can increase the HAM value.

Considering only jumps we can prove an upper bound of $O(n^4 \log n)$, since two special jumps can simulate an exchange step. However, the probability of choosing these two jumps is bounded by $O(1/n^4)$. A lower bound for jump operations has to take into account that a single jump $((2, 3, \ldots, n, 1) \to (1, 2, 3, \ldots, n))$ can increase the HAM value from 0 to n. However, we do not believe that jumps help for this fitness landscape. Hence, for HAM exchanges play the role which is played by jumps for REM.

Theorem 4. *The expected optimization time of the* $(1 + 1)$*EA on the fitness landscape described by EXC is bounded above by* $O(n^3)$.

Proof. First, $\mathrm{EXC}(\pi) \leq n-1$ for all π. It is always possible to choose an accepted exchange operation increasing the HAM value by 1. Since $\mathrm{HAM}(\pi) \neq n - 1$ for all π, the statement follows. Hence, the expected optimization time to decrease the EXC value is $O(n^2)$ leading to the proposed bound. □

Since two special jumps can simulate a given exchange step, we obtain for jumps an $O(n^5)$ bound. It is not surprising that exchanges seem to be better than jumps for this fitness function which is defined via the minimal number of exchange operations.

All the fitness functions based on INV, REM, HAM, or EXC lead to fitness landscapes which are easy for simple EAs. The expected optimization time can depend essentially on the chosen mutation operator. Hence, it seems to be useful to allow jumps and exchanges. However, we still have to investigate the fitness landscape based on the number of runs (RUN). This is the perhaps best-known measure of presortedness, since Mergesort is based on the idea of reducing the number of runs efficiently.

Our aim is to prove that our EAs have an exponential expected optimization time on this landscape. We start with a well-structured subcase. The current search point π consists of two runs of length $k \leq n/2$ and $n - k$. The k elements of the first run are chosen randomly from $\{1, \ldots, n\}$. In the case that these are the elements $1, \ldots, k$ (probability $1/\binom{n}{k}$) or $k = 0$ the search is finished successfully. We first consider the mutation operator which performs exactly one jump per step. One element i from one run is chosen randomly for the jump. Most jump destinations increase the number of runs. In general, there is exactly one position j in the other run such that $\mathrm{jump}(i, j)$ reduces the length of the chosen run by 1 and increases the length of the other run by 1. The set of elements in each run is still a random subset of $\{1, \ldots, n\}$ of the correct size. If we consider t steps

where the runs always have a length of at least k^*, the probability of finishing the search is bounded above by $t/\binom{n}{k^*}$. If $k^* \geq n^{1/2}$, the probability to finish the search within $2^{cn^{1/2}}$ steps, $c > 0$ small enough, is bounded by $2^{-\Omega(n^{1/2})}$. There is one special situation, namely $1, \ldots, k-1, j, k, \ldots, j-1, j+1, \ldots, n$. If we choose the element j for a jump, it may jump to position $k+1$ and we get the runs $1, \ldots, k, j$ and $k+1, \ldots, j-1, j+1, \ldots, n$. This is identical to a jump of element k to its correct position k in the other run. Also this special situation has a vanishing probability if $k^* \geq n^{1/2}$. The probability of a successful step equals $1/(n-1)$. Hence, we decrease the expected optimization time by a factor of $1/(n-1)$ by considering only successful steps. The probability of increasing k, the length of the shorter run, during a successful step equals k/n (we have to choose the jumping element from the shorter run) while the probability of increasing k equals $(n-k)/n$. In order to finish the search we have to decrease k to 0, but k has a strong tendency to be increased if k gets small.

Let k be the current length of the shorter run. We try to estimate the probability of reaching the value 0 before the value $2k$. Therefore, it is necessary to have at least k decreasing steps among the next $3k$ steps. The value $p = 2k/n$ is the largest "decrease probability" as long as the length of the shorter run is at most $2k$. Then the considered probability can be estimated by

$$\sum_{k \leq m \leq 3k} \binom{3k}{m} p^m (1-p)^{3k-m} \leq \binom{3k}{k} p^k (1 + 3p + 9p^2 + \cdots),$$

since the binomial coefficients $\binom{3k}{m}$ are at most increasing by a factor of 3, if $m \geq k$. Moreover, $\binom{3k}{k} \leq c(3/2)^k$. If $k \leq n/8$, $3p = 6k/n \leq 3/4$ and we can estimate the considered probability by $4c(3k/n)^k$ which decreases exponentially with k. If k is the length of the shorter run in the beginning, we need once a phase starting with the value k and at least k decreasing steps among $3k$ steps. The expected waiting time is $\Omega((n/(3k))^k)$ and the success probability within t steps can be bounded above by $O(t(3k/n)^k)$. If k is not too small, e.g., $k = n^{1/2}$, we only have an exponentially small probability to finish the search within a period of time which grows exponentially but not too fast.

Single exchange steps are almost useless. The exchange of the last element of the first run and the first element of the second run is also a jump and may change the lengths of the runs. All other accepted exchange steps exchange two elements from both runs which does not change the lengths of the runs.

What is the effect of many jumps within an operation? With overwhelming probability we do not perform more than $n^\varepsilon/4$ jumps in one step. This holds for the Poisson distribution for each $\varepsilon > 0$. Therefore, an element jumping to a position with a distance of more than $n^\varepsilon/4$ from each of the two correct positions in the runs has to jump again such that we do not obtain more than two runs. If we choose r different elements which have to jump, the probability that the resulting sequence has at most two runs is bounded above $(n^\varepsilon/n)^r = n^{-(1-\varepsilon)r}$ while this probability is at least $1/(n-1)$ for $r = 1$. Since the number of jumps equals $X + 1$ where X has a Poisson distribution with $\lambda = 1$, the probability

of letting at least r elements jump is bounded by $(2/(r-1)!)$. Hence, we may assume that all successful steps where $r \geq 2$ elements jump shorten the shorter run by r. Since the expected decrease of the shorter run during $3k$ steps caused by this assumption is only $O(kn^{-(1-\varepsilon)\cdot 2})$, we can generalize our estimates above (with minor changes) to obtain the same results for the (1+1)EA.

The last question is whether it is likely that we reach a situation with two runs where the shorter one has a length of at least $n^{1/2}$ and where the probability that this run contains almost only very small or almost only very large elements is tiny. In each situation it is more likely to increase the length of a long run than to increase the length of a short run. However, for short runs there is a non-vanishing probability of merging runs. Altogether, it seems to be very likely that we reach a situation as described above. However, this has not been proved rigorously here. The result proven rigorously is stated as a theorem.

Theorem 5. *Let $n^{1/2} \leq k \leq n/2$ and let π be chosen randomly among all sequences with two runs of length k and $n-k$ resp. The expected optimization time of the (1+1)EA on the fitness landscape described by RUN and the random initial string π is bounded below by $2^{\Omega(n^{1/2})}$ and the success probability within $2^{cn^{1/2}}$ steps, $c > 0$ a small constant, is exponentially small.*

Our experiments have confirmed our statements discussed above. For $20 \leq n \leq 40$, in almost all cases the length of the shorter run was at least $0.4n$ for the first string with two runs. For $n = 40$, we have obtained the first string with two runs on the average only after approximately $9 \cdot 10^7$ steps. For $n = 60$ this happens only after $18 \cdot 10^7$ steps and the optimum is not reached after $5.7 \cdot 10^9$ steps. The fitness landscape based on RUN is difficult already for quite small n.

4 Fitness Landscapes Based on Shortest Paths Problems

Given n nodes and a distance matrix with positive entries d_{ij} the single-source-shortest-paths problem (SSSP) is the problem to compute for each node $v \neq s$ a shortest path from node s to node v. Using Dijkstra's algorithm SSSP can be solved in time $O(n^2)$. It seems to be a good idea to consider the search space of all $v = (v_1, \ldots, v_{n-1}) \in \{1, \ldots, n\}^{n-1}$ where $v_i \neq i$ with the following interpretation. We set $s = n$ and obtain a graph where i has the single predecessor v_i. If this graph has a cycle, the individual is invalid with fitness ∞. Otherwise, we get a tree rooted at s describing for each $i \neq s$ an s-i-path. As single-objective problem the fitness equals the sum of the lengths of all s-i-paths. This may lead to a needle-in-the-haystack landscape. If $d_{i,i-1} = 1$ and $d_{ij} = \infty$, if $j \neq i-1$, only the optimal tree where $v_i = i+1$ has a fitness smaller than ∞.

Now we consider SSSP as a multi-objective optimization problem where we have $n-1$ objectives namely the lengths of the s-i-paths. We use the same search space with the fitness function $f(v) = (f_1(v), \ldots, f_{n-1}(v))$ where $f_i(v)$ is the length of the s-i-path in the graph described by v, if this graph contains such a path, and $f_i(v) = \infty$, otherwise. The aim is minimization. The theory on shortest

paths problems implies that this multi-objective minimization problem has a unique Pareto optimal fitness vector $f^* = (l_1^*, \ldots, l_{n-1}^*)$ containing the lengths of shortest s-i-paths and all Pareto optimal vectors describe graphs representing a system of shortest paths. (A vector is called Pareto optimal if it is minimal w.r.t. the partial order $\leq$ on $(\mathbb{R} \cup \{\infty\})^{n-1}$ where $(a_1, \ldots, a_{n-1}) \leq (b_1, \ldots, b_{n-1})$ iff $a_i \leq b_i$ for all i.)

We may design variants of evolutionary algorithms for multi-objective minimization. However, a very simple variant again called (1+1)EA is an efficient SSSP solver. We use the following mutation operator:

- Choose s according to a Poisson distribution with parameter $\lambda = 1$ and perform sequentially $s + 1$ flips. A flip chooses randomly a position $p \in \{1, \ldots, n-1\}$ and replaces v_p by a random node $w \neq p$.

The (1+1)EA chooses the first search point v randomly, creates v' by the mutation operator and replaces v by v' iff $f(v') \leq f(v)$.

Theorem 6. *The expected optimization time of the multi-objective (1+1)EA on SSSP is bounded above by $O(n^3)$.*

We prove a more sophisticated bound. Let t_i be the smallest number of edges on a shortest s-i-path, $m_j := \#\{i \mid t_i = j\}$, and $T = \max\{j \mid m_j > 0\}$. Then we prove the upper bound

$$en^2 \sum_{1 \leq j \leq T} (\ln m_j + 1).$$

This bound has its maximal value $\Theta(n^3)$ for $m_1 = \cdots = m_{n-1} = 1$. We also obtain the bound $O(n^2 T \log n)$ which in the typical case where T is small is much better than $O(n^3)$.

Proof. The proof is based on the following simple observation. Whenever $f_i(v) = l_i^*$, we only accept search points v' where $f_i(v') = l_i^*$. Hence, we do not forget the length of shortest paths which we have found (although we may switch to another shortest path). Now we assume that we have a search point v where $f_i(v) = l_i^*$ for all i where $t_i < t$. Then we wait until this property holds for all i where $t_i \leq t$. For each node i where $t_i = t$ and $f_i(v) > l_i^*$ there exists a node j such that $t_j = t - 1$, j is the predecessor of i on a shortest s-i-path of length t, and $f_j(v) = l_j^*$. Then a mutation where only v_i is flipped and obtains the value j is accepted and leads to a search point v' where $f_i(v') = l_i^*$. The probability of such a mutation equals $1/(e(n-1)^2)$ ($1/e$ the probability of flipping exactly one position, $1/(n-1)$ the probability of flipping the correct position, and $1/(n-1)$ the probability of flipping it to the right value). If we have r such nodes, the success probability is at least $r/(e(n-1)^2)$ and the expected waiting time is bounded above by en^2/r. The largest value for r is m_t and we have to consider each of the values $m_t, \ldots, 1$ at most once. Hence, the total expected time of this phase is bounded above by $en^2(1 + \frac{1}{2} + \cdots + \frac{1}{m_t}) \leq en^2(\ln m_t + 1)$. Since t can take the values $1, \ldots, T$ we have proved the proposed bound. □

It is easy to show that the expected optimization time equals $\Theta(n^3)$ in the extreme case which is a needle-in-the-haystack landscape for single-objective optimization.

5 Conclusion

Robust problem solvers should also solve the well-known simple optimization problems efficiently. This has been investigated for the sorting problem (maximizing the sortedness based on some measure of presortedness) and shortest-paths problems. For four out of five fitness landscapes described by the best-known measures of presortedness simple EAs work very efficiently, although different types of local changes are essential. However, the last measure of presortedness leads to a fitness landscape which is difficult for EAs. There are instances of shortest-paths problems which are difficult for black-box single-objective optimization. The modeling of the SSSP as multi-objective optimization problem reflects the structure of the problem and the fitness vector reveals enough information to direct the search of a simple EA. Usually, multi-objective optimization is only used if no single-objective optimization problem contains the whole structure of the problem. Here it has been shown that a multi-objective problem model may lead to a simpler problem.

References

1. Bäck, T., Fogel, D.B., and Michalewicz, Z. (Eds.) (1997). *Handbook of Evolutionary Computation*. Oxford University Press.
2. Droste, S., Jansen, T., and Wegener, I. (1998a). A rigorous complexity analysis of the (1+1) evolutionary algorithm for separable functions with Boolean inputs. Evolutionary Computation 6, 185–196.
3. Droste, S., Jansen, T., and Wegener, I. (1998b). On the optimization of unimodal functions with the (1+1) evolutionary algorithm. Parallel Problem Solving from Nature – PPSN V, LNCS 1498, 13–22.
4. Horn, J., Goldberg, D.E., and Deb, K. (1994). Long path problems. Parallel Problem Solving from Nature – PPSN III, LNCS 866, 149–158.
5. Jansen, T., and Wegener, I. (2001a). Real royal road functions - where crossover provably is essential. Genetic and Evolutionary Computation Conf. – GECCO, 375–382.
6. Jansen, T., and Wegener, I. (2001b). Evolutionary algorithms - how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Trans. on Evolutionary Computation. To appear Dec. 2001.
7. Mitchell, M., Holland, J.H., and Forrest, S. (1994). When will a genetic algorithm outperform hill climbing. In J. Cowan, G. Tesauro, and J. Alspector (Eds.): *Advances in Neural Information Processing Systems*. Morgan Kaufman.
8. Petersson, O., and Moffat, A. (1995). A framework for adaptive sorting. Discrete Applied Mathematics 59, 153–179.
9. Rudolph, G. (1997). How mutations and selection solve long path problems in polynomial expected time. Evolutionary Computation 4, 195–205.
10. Wegener, I. (2001). Theoretical aspects of evolutionary algorithms. Int. Colloq. on Automata, Languages, and Programming – ICALP, LNCS 2076, 64–78.