An Analysis on Recombination in Multi-Objective Evolutionary Optimization*

Chao Qian Yang Yu Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China
{qianc,yuy,zhouzh}@lamda.nju.edu.cn

ABSTRACT

Recombination (or called *crossover*) operators are a kind of characterizing feature of evolutionary algorithms (EAs). The usefulness of recombination operators has been verified empirically in many practical applications, and has also been theoretically studied in single-objective optimization. For multi-objective optimization, however, there lacks strong evidence on whether the recombination operators can lead to a better running time. In this paper, we establish some theoretical support to the use of recombination in multi-objective optimization. We analyze the running time of REMO, a simple multi-objective EA with a recombination operator, on two well-studied bi-objective problems, i.e., the LOTZ and the COCZ problems. Our analysis results disclose that the average running time of REMO on LOTZ and COCZ is $\Theta(n^2)$ and $\Theta(n \log n)$, respectively, improved from $\Theta(n^3)$ and $\Theta(n^2)$ as when the recombination operator is turned off, respectively. These results imply that the recombination operator is crucial for the efficiency of REMO on these two problems. The analysis also suggests that, generally, recombination operators can be helpful to multi-objective optimization as they may accelerate the filling of the Pareto front through recombining diverse solutions.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Theory

Keywords

Evolutionary algorithms, multi-objective optimization, recombination, crossover

*This research was supported by the National Science Foundation of China (61073097), the Jiangsu Science Foundation (BK2008018) and the National Fundamental Research Program of China (2010CB327903).

GECCO'11, July 12–16, 2011, Dublin, Ireland.

1. INTRODUCTION

In evolutionary algorithms (EAs) [1], recombination (or called *crossover*) operators are used to generate offspring solutions from existing ones, they take two or more individual solutions from the population pool maintained by an EA and exchange some parts of the solutions to form new solutions. Recombination operators are a kind of characterizing feature of evolutionary algorithms [1]. In recent development of multi-objective optimization, recombination operators are also used in multi-objective EAs, from early work (e.g., [21]) to up-to-date work (e.g., [28, 24]). However, there are few theoretical results on whether recombination operators are useful for EAs addressing multi-objective problems, and how much useful they can be. The answers to these questions can enhance our understanding of this kind of nature-inspired operators and may disclose potential for further developments.

1.1 Related Work

Many theoretical results on the running time of EAs have been derived and approaches to analyze EAs have been developed, e.g. [6, 13, 36]. Most of these analysis focus on EAs with only mutation operators, while only a few touch recombination operators.

Properties of recombination operators have been addressed in the scenario of single-objective optimization. Early empirical analysis includes [20] and [32]. Running time analysis later provided theoretical understanding of recombination operators. Several crossover-only evolutionary algorithms were proved to be effective on the H-IFF problem which is hard for any kind of mutation-based EAs [34, 2]. Jansen and Wegener [15, 16] proved that crossover operators can be crucially important on some artificial problems. Crossover operators then have been proved to be useful in more cases, including Ising models [8, 33], the TwoPaths instance class of the problem of computing unique input-output sequences [19], some instance classes of the vertex cover problem [26], and the all-pairs shortest path problem [3, 5, 4]. Meanwhile, on the contrary, Richter et al. [27] gave the Ignoble Trail functions where a crossover operator was proved to be harmful. Approaches of analysis have been developed such as the *Markov chain switching theorem* [35] that compares the running time of an EA turning recombination on and off. While all these studies are in the scenario of single-objective optimization, the results can not be easily generalized to the scenario of multi-objective optimization. Particularly, multi-objective optimization aims at finding a set of optimal and non-dominant solutions rather than a single optimal solution, where the situation becomes more complex and is untouched.

Early analysis of multi-objective EAs (MOEAs) concentrates on the conditions under which MOEAs can find the global optimal solution given unlimited time, e.g. [11, 12, 29, 30, 31]. For running time analysis, studies on two bi-objective pseudo-boolean func-

Table 1: Comparison of expected running time of several multi-objective EAs on LOTZ and COCZ.

Problem	(1+1)-EMO	SEMO	FEMO	GEMO	$\text{REMO}_{0.5}$	REMO_0
LOTZ	$\Theta(n^3)$ [17]	$\Theta(n^3)$ [18]	$\Theta(n^2 \log n)$ [18]	$\Theta(n^2 \log n)$ [17]	$\Theta(n^2)$	$\Theta(n^3)$
COCZ	$\Theta(n^2 \log n)$ [17]	$O(n^2 \log n)$ [17]	$O(n^2 \log n)$ [17]	$\Theta(n^2)$ [17]	$\Theta(n \log n)$	$\Theta(n^2)$

tions, the LOTZ and the COCZ problems extended respectively from the well studied LeadingOnes and OneMax problems, have led to some disclosure of limited time behaviors of MOEAs. Table 1 lists the analysis results of several MOEAs on the two problems. Note that none of these previously analyzed MOEAs uses recombination operators.

The recent work by Neumann and Theile [24] is, to the best of our knowledge, the first and only work analyzing crossover operators in MOEAs. They proved that a crossover operator can speed up evolutionary algorithms for the multi-criteria all-pairs-shortest-path problem. Discovered through their analysis, the crossover operator can be helpful in the interplay with the mutation operator, such as that the crossover operator can generate a good starting point solution from which the mutation operator can evolve the population efficiently.

We also note that there are a bunch of studies that analyze MOEAs for solving single-objective problems [25, 22, 9, 23, 14]. However, we concern about multi-objective problems.

1.2 Our Contribution

As that multi-objective optimization aims at finding a set of optimal solutions with different balances of the objectives, which was not involved in the analysis for single objective optimization, this paper investigates whether recombination operators can have any effect on solving the optimal solution set.

We study the performance of REMO (recombination-incorporated evolutionary multi-objective optimizer), which is extended from the SEMO [18] algorithm, on the LOTZ and the COCZ problems. We prove that the average running time of REMO using recombination with probability 0.5 (denoted as $\text{REMO}_{0.5}$) is $\Theta(n^2)$ on the LOTZ problem, and $\Theta(n \log n)$ on the COCZ problem. Meanwhile, we also analyze REMO with the recombination operator turned off (denoted as REMO_0), and prove that the average running time of REMO_0 is $\Theta(n^3)$ on the LOTZ problem and $\Theta(n^2)$ on the COCZ problem. Comparing the analysis results of $\text{REMO}_{0.5}$ with REMO_0, we conclude that the recombination operator is crucial for the efficiency of REMO on the two problems. The running time of REMO and all the other known results is compared in Table 1. It can be observed that, with the help of the recombination operator, $\text{REMO}_{0.5}$ has the best running time on the LOTZ problem, and has better running time than (1+1)-EMO and GEMO on the COCZ problem while has better upper bound than SEMO and FEMO.

Through the analysis of REMO on the two problems, we discover that the recombination operator can work by accelerating the filling of the Pareto front through recombining diverse optimal solutions that have been found. It is worth noticing that this mechanism is different from that analyzed in [24], where the crossover operator works as its interplay with mutation. Moreover, the idea that recombination can accelerate the filling of the Pareto front through recombining diverse solutions, though only proved in the specific cases, might be hold in more general situations and might be useful for designing more efficient MOEAs.

The rest of this paper is organized as follows. Section 2 introduces the preliminaries on multi-objective optimization. Section 3 introduces the model problems. Section 4 presents the REMO algorithm, which is analyzed in Section 5. Section 6 concludes.

2. MULTI-OBJECTIVE OPTIMIZATION

Multi-objective optimization requires to simultaneously optimize two or more objective functions, as in Definition 1. When there are two objective functions, it is also called as a *bi*-objective optimization problem. We consider maximization problems in this paper.

Definition 1 (Multi-Objective Optimization) *Given a feasible solution space $\mathcal{X}$ and objective functions $f_1, \ldots, f_m$, the maximum multi-objective optimization aims to find the solution $\boldsymbol{x}^*$ satisfying*

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \boldsymbol{f}(\boldsymbol{x}) = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \big(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})\big)$$

where $\boldsymbol{f}(\boldsymbol{x}) = \big(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})\big)$ is the objective vector of the solution $\boldsymbol{x}$.

Usually, the objectives are conflicted, i.e., optimization of one objective alone will degrade the other objectives, and it is impossible to have one solution that optimizes all the objectives. Therefore, multi-objective optimization tries to find a set of solutions according to some criterions. One commonly used criterion is the Pareto optimality, which utilizes the *domination* relation between solutions as in Definition 2. The solution set by Pareto optimality is called Pareto set, as in Definition 3.

Definition 2 (Domination) *Let $\boldsymbol{f} = (f_1, f_2, \ldots, f_m) : \mathcal{X} \to \mathbb{R}^m$ be the objective vector, where $\mathcal{X}$ is the feasible solution space, and $\mathbb{R}^m$ is the objective space. For two solutions $\boldsymbol{x}$ and $\boldsymbol{x}' \in \mathcal{X}$:*

1. *$\boldsymbol{x}$* weakly dominates *$\boldsymbol{x}'$ if, for all i that $1 \leq i \leq m$, $f_i(\boldsymbol{x}) \geq f_i(\boldsymbol{x}')$, denoted as $\succeq_{\boldsymbol{f}}$;*
2. *$\boldsymbol{x}$* dominates *$\boldsymbol{x}'$ if, $\boldsymbol{x}$ weakly dominates $\boldsymbol{x}'$ and $f_i(\boldsymbol{x}) > f_i(\boldsymbol{x}')$ for some i, denoted as $\succ_{\boldsymbol{f}}$.*

Definition 3 (Pareto Optimality) *Let $\boldsymbol{f} = (f_1, f_2, \ldots, f_m) : \mathcal{X} \to \mathbb{R}^m$ be the objective vector, where $\mathcal{X}$ is the feasible solution space, and $\mathbb{R}^m$ is the objective space. A solution $\boldsymbol{x} \in S$ is* Pareto optimal *if there is no other solution in $\mathcal{X}$ that dominates $\boldsymbol{x}$. A set S is called* Pareto set *if it contains only Pareto optimal solutions. The collection of objective values of a Pareto set is called the Pareto front of the set.*

With the goal of finding the largest Pareto set, or called the *optimal Pareto set*, the running time of an MOEA is counted as the number of calls to $\boldsymbol{f}$ until it finds the Pareto front of the optimal Pareto set, that is, the MOEA should find at least one corresponding solution for each element in the Pareto front of the optimal Pareto set. Note that this definition agrees with that in [18, 10, 17]. Since MOEAs are naturally stochastic algorithms, we measure the performance of MOEAs by the expected running time. SEMO [18], as described in Algorithm 1, is a simple MOEA, and also the first analyzed MOEA due to its simplicity, which explains the common structure of various MOEAs.

Algorithm 1 (SEMO [18]) *Given solution length n and objective function vector $\boldsymbol{f}$, SEMO consists of the following steps:*

1: *Randomly choose* $\boldsymbol{x} \in \{0,1\}^n$
2: $P \leftarrow \{\boldsymbol{x}\}$
3: ***loop***
4: *Choose* $\boldsymbol{x}$ *from* P *uniformly at random*
5: *Create* $\boldsymbol{x}'$ *by flipping a randomly chosen bit of* $\boldsymbol{x}$
6: $P \leftarrow P - \{\boldsymbol{z} \in P | \boldsymbol{x}' \succ_{\boldsymbol{f}} \boldsymbol{z}\}$
7: ***if*** $\nexists \boldsymbol{z} \in P$ *such that* $\boldsymbol{z} \succeq_{\boldsymbol{f}} \boldsymbol{x}'$ ***then***
8: $P \leftarrow P \cup \{\boldsymbol{x}'\}$
9: ***end if***
10: ***end loop***

3. THE MODEL PROBLEMS

Two bi-objective pseudo-boolean model problems, LOTZ (Leading Ones Trailing Zeros) and COCZ (Count Ones Count Zeros), are usually used to investigate the properties of MOEAs [18, 10, 17], as the listed in Table 1.

For LOTZ, the first objective is to maximize the number of leading one bits (the same as the LeadingOnes problem [6]), and the other objective is to maximize the number of trailing zero bits.

Definition 4 (LOTZ) *The Pseudo-Boolean function* $LOTZ$*:* $\{0,1\}^n \to \mathbb{N}^2$ *is defined as follows:*

$$LOTZ(\boldsymbol{x}) = \left(\sum_{i=1}^{n} \prod_{j=1}^{i} x_j, \sum_{i=1}^{n} \prod_{j=i}^{n} (1 - x_j) \right).$$

As analyzed in [18], the objective space of LOTZ can be partitioned into $n+1$ subsets F_i, where $i \in \{0, \ldots, n\}$ is the sum of the two objective values, i.e., $\boldsymbol{f}(\boldsymbol{x}) \in F_i$ if $f_1(\boldsymbol{x}) + f_2(\boldsymbol{x}) = i$. Obviously, F_n is the Pareto front of the optimal Pareto set, and the optimal Pareto set has $n+1$ elements, which are $0^n, 10^{n-1}, \ldots, 1^n$.

For COCZ, the first objective is to maximize the number of one bits (the same as the OneMax problem [6]), and the other objective is to maximize the number of one bits in the first half of the solution plus the number of zero bits in the second half. The two objectives are corporative in maximizing the number of one bits in the first half of the solution, but conflict in the second half.

Definition 5 (COCZ) *The Pseudo-Boolean function* $COCZ$*:* $\{0,1\}^n \to \mathbb{N}^2$ *is defined as follows:*

$$COCZ(\boldsymbol{x}) = \left(\sum_{i=1}^{n} x_i, \sum_{i=1}^{n/2} x_i + \sum_{i=n/2+1}^{n} (1 - x_i) \right)$$

where n is even.

As analyzed in [17], the objective space of COCZ can be partitioned into $n/2+1$ subsets F_i, where $i \in \{0, \ldots, n/2\}$ is the number of one bits in the first half of the solution. It is obvious that each F_i contains $n/2+1$ different objective vectors $(i+j, i+n/2-j)$, where $j \in \{0, \ldots, n/2\}$ is the number of one bits in the second half. The Pareto front of the optimal Pareto set is $F_{n/2}$, and the optimal Pareto set is $\{1^{\frac{n}{2}} * ^{\frac{n}{2}}; * \in \{0,1\}\}$, the size of which is $2^{\frac{n}{2}}$.

4. REMO

The recombination-incorporated evolutionary multi-objective optimizer (REMO) studied in this paper is depicted in Algorithm 2, which is extended from the algorithm SEMO by incorporating a recombination operator. The components of REMO is explained in the following.

It is well known that the diversity of the population is important to the success of recombination operators, since recombination makes no progress from similar solutions. Therefore, we should employ some diversity control in REMO. We use *objective diversity (od)* measure with an assumption that the diversity of a set of solutions is consistent with the difference of their objective vectors. By this assumption, a population containing solutions with good objective values on different objectives has a high diversity. The objective diversity is defined as follows. For a set of solutions S, define a variable q_i for the i-th objective $(1 \leq i \leq m)$,

$$q_i = \begin{cases} 1 & \text{if } \max\{f_i(\boldsymbol{x}) \mid \boldsymbol{x} \in S\} \geq \theta_i, \\ 0 & \text{otherwise,} \end{cases}$$

where θ_i is the "goodness" threshold of the i-th objective, then we define the objective diversity of S

$$od(S) = \sum_{i=1}^{m} q_i.$$

That is, the objective diversity of a population is the number of objectives that have corresponding good solutions in the population, and a solution is good for an objective if its objective value is not less than a predefined threshold. Then, given m objectives, the largest value of objective diversity is m. Here, we use the objective diversity with $\theta_i =$ the minimal local optimal value of the i-th objective.

To make the initial population diverse enough, we use an initialization process, described in Subprocedure 1. In the initialization process, m independent randomized local search (RLS) are employed to optimize the m objective functions $f_1, f_2, \ldots, f_m$, each RLS corresponds to one objective. In the RLS, the solution is examined whether it is local optimal for an objective every n mutation steps, where n is the length of the solution. When to check whether a solution is local optimal for an objective, we use the solutions having Hamming distance 1 with the current solution as the neighborhood. This initialization procedure is terminated when local optimal solutions are found for all the objectives.

Subprocedure 1 (Initialization) *Input: m solutions $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m$ from $\{0,1\}^n$; Output: m local optimal solutions corresponding to one of the m objectives respectively; the Initialization procedure consists of the following steps:*

1: ***repeat***
2: ***repeat*** *the following process n times*
3: *Create offsprings $\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_m$ by flipping a randomly chosen bit of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m$ respectively*
4: ***if*** $f_i(\boldsymbol{x}'_i) > f_i(\boldsymbol{x}_i)$ *for an* i ***then***
5: $\boldsymbol{x}_i \leftarrow \boldsymbol{x}'_i$
6: ***end if***
7: ***end repeat***
8: ***until*** $\forall i$: $\boldsymbol{x}_i$ *is a local optima for* f_i

The process makes the initial population contain one good solution for each objective, i.e., $q_i = 1 (1 \leq i \leq m)$. Thus, the objective diversity of the initial population is m. Since a solution in the population is eliminated only if it is dominated by a new solution, there always exists good solutions for each objective. As the result, the objective diversity of the population always keeps m, the maximal objective diversity, throughout the evolutionary process.

In each reproduction step, REMO picks a set of m solutions with the objective diversity at least $m/2$ from the current population to carry out the recombination. In order to do so, the best solutions each for one of the randomly selected $m/2$ objectives are selected at first, denoted as a set P_s^1. The remaining $m - |P_s^1|$ solutions

are randomly chosen from the population excluding P_s^1. Thus, the selected solutions for recombination have an objective diversity at least $m/2$.

In the reproduction procedure, we use the parameter p_c to control the use of the crossover, that is, in each reproduction step, the offspring solutions are generated by the crossover with probability p_c, and otherwise generated by the mutation. In the end of each iteration, the offspring solutions P_o are used to update the current population.

Algorithm 2 (REMO) *Given solution length n and objective function vector $\boldsymbol{f}$ of length m, REMO consists of the following steps:*

1: $P \leftarrow$ *Initialization*(m *solutions randomly from* $\{0,1\}^n$)
2: ***loop***
3: $P_s^1 \leftarrow$ *select the best solution in* P *for each of the randomly selected* $m/2$ *objectives*
4: $P_s^2 \leftarrow$ *randomly select* $m - |P_s^1|$ *solutions from* $P - P_s^1$
5: $P_s \leftarrow P_s^1 \cup P_s^2$
6: $r \leftarrow$ *uniformly chosen from* $[0,1]$ *at random*
7: ***if*** $r < p_c$ ***then***
8: $P_o \leftarrow$ *Recombination*(P_s)
9: ***else***
10: $P_o \leftarrow$ *for each solution* $\boldsymbol{x} \in P_s$, *flip a randomly chosen bit*
11: ***end if***
12: ***for*** *each solution* $\boldsymbol{x}' \in P_o$
13: $P \leftarrow P - \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \succ_{\boldsymbol{f}} \boldsymbol{z}\}$
14: ***if*** $\nexists \boldsymbol{z} \in P$ *such that* $\boldsymbol{z} \succeq_{\boldsymbol{f}} \boldsymbol{x}'$ ***then***
15: $P \leftarrow P \cup \{\boldsymbol{x}'\}$
16: ***end if***
17: ***end for***
18: ***end loop***

The recombination operator employed in REMO is the diagonal multi-parent crossover [7], as in Definition 6. For m solutions, diagonal multi-parent crossover randomly selects $m-1$ crossover points between adjacent bits and creates m offspring solutions by sequentially combining the components partitioned by the crossover points, which is a generalization of one-point crossover over two solutions.

Definition 6 (Recombination [7]) *Given m solutions whose length is n, randomly select $m-1$ crossover points from $n-1$ positions between adjacent bits, and create m offspring solutions as follows. Denote the order of the m parents as $1, 2, ..., m$. The m offspring solutions are generated by combining m components partitioned by the $m-1$ crossover points, where the components of the ith offspring solution sequentially come from parents $i, i+1, ..., m-1, m, 1, ..., i-1$, where $1 \leq i \leq m$.*

Note that we investigate the bi-objective problems in this paper, thus when selecting solutions for reproduction, the best solution in the current population for a randomly selected objective is selected first, and then the other solution is randomly chosen from the remaining solutions. Also since the bi-objective problems are considered in this paper, the recombination operator used in REMO is just one-point crossover.

5. ANALYSIS OF REMO

In this section, we prove the running time of REMO on the LOTZ problem as well as on the COCZ problem. First, we analyze the running time of the initialization procedure of REMO on these two problems.

Lemma 1 *On the* LOTZ *problem, the expected running time of the initialization procedure of REMO is $\Theta(n^2)$.*

Proof. The initialization procedure is to optimize the two objectives of LOTZ separately by two independent RLS. For the first objective LO which is to maximize the number of leading ones, the probability of increasing LO in one mutation step is $1/n$ and the expected number of 0's of the initial random solution is $n/2$, thus, the expected running time to find the solution optimizing LO is $n^2/2$. Meanwhile, a solution is examined whether it is local optimal for an objective every n mutation steps, thus it needs to examine $n/2$ times in expectation in the evolutionary process. Since the neighborhood size of a solution is n, the running time of checking whether a solution is local optimal is n. Thus, the expected running time to find the optimal solution for the objective LO is $n^2/2 + n \cdot n/2 = n^2$. The result also holds for optimizing the second objective TZ. Thus, the expected running time of the initialization procedure of REMO on LOTZ is $\Theta(n^2)$. ■

Lemma 2 *On the* COCZ *problem, the expected running time of the initialization procedure of REMO is $\Theta(n \log n)$.*

Proof. The initialization procedure is to optimize the two objectives of COCZ separately by two independent RLS. For the first objective which is to maximize the number of 1's, the probability of increasing the objective by 1 in one mutation step from a solution with i 0's is $\frac{i}{n}$. Thus, the expected running time for a solution with i 0's is nH_i. Since a random solution uniformly selected from $\{0,1\}^n$ has i 0's with probability $\binom{n}{i}/2^n$, the expected running time for optimizing the first objective is $\sum_{i=0}^{n}\binom{n}{i}nH_i/2^n \in [\frac{n}{2}H_{\frac{n}{2}}, nH_{\frac{n}{2}}]$. Meanwhile, a solution is examined whether it is local optimal for an objective every n mutation steps, then it needs to examine $[\frac{1}{2}H_{\frac{n}{2}}, H_{\frac{n}{2}}]$ times in expectation in the evolutionary process, where the cost of one examination is n. Thus, the expected running time to find a local optima for the first objective is $[nH_{\frac{n}{2}}, 2nH_{\frac{n}{2}}] \in \Theta(n \log n)$. The result also holds for optimizing the second objective. Thus, the expected running time of the initialization procedure of REMO on COCZ is $\Theta(n \log n)$. ■

Then, we derive a property of the evolving population of REMO on the LOTZ problem as well as on the COCZ problem.

Lemma 3 *On the* LOTZ *problem, after the initialization procedure, the population of REMO contains only Pareto optimal solutions and its size never decreases.*

Proof. After the initialization procedure, the solutions 1^n and 0^n are generated, which are local optima for the objective LO and TZ, respectively. Thus, the initial population $P = \{1^n, 0^n\}$ contains two Pareto optimal solutions. Assume that the current population contains only Pareto optimal solutions. In the next step, after using either one-point crossover or one bit mutation on the two selected Pareto optimal solutions, the offspring solutions which are not Pareto optimal will be dominated by the parents, and only the new Pareto optimal solutions will be accepted. Thus, the next population also contains only Pareto optimal solutions. Then, it is obvious that the size of the population never decreases. ■

Lemma 4 *On the* COCZ *problem, after the initialization procedure, the population of REMO contains only Pareto optimal solutions and its size never decreases.*

Proof. After the initialization procedure, the solutions 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$ are generated, which are local optima for the two objectives of COCZ, respectively. Thus, the initial population $P = \{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$

contains two Pareto optimal solutions. Assume that the current population contains only Pareto optimal solutions. In the next step, on the two selected Pareto optimal solutions, if it uses one-point crossover, only Pareto optimal solutions can be generated; otherwise it uses mutation, the offspring solutions which are not Pareto optimal are dominated by the parents. Thus, the next population also contains only Pareto optimal solutions. Then, it is obvious that the size of the population never decreases. ■

5.1 Analysis of REMO with $p_c = 0.5$

Theorem 1 *On the* LOTZ *problem, the expected running time of* REMO *with* $p_c = 0.5$ *is* $\Theta(n^2)$.

Proof. For the LOTZ problem, the size of the Pareto front of the optimal Pareto set is $n + 1$ and the Pareto optimal solution has a one-to-one correspondence with its objective vector. By Lemma 3, we divide the optimization process into $n - 1$ phases, where the i-th phase $(1 \leq i \leq n - 1)$ corresponds to the process of evolving the population containing $i + 1$ Pareto optimal solutions to that containing $i + 2$ Pareto optimal solutions, and then bound the running time of the whole evolutionary process by summing up the running time bound of each phase.

We consider the i-th phase where the population contains $i + 1$ Pareto optimal solutions. In the selection procedure of REMO, the best solution in the current population for a randomly chosen objective will be selected first, then the other solution will be randomly selected from the remaining solutions. Since the two solutions 1^n and 0^n, which are optimal for the objectives LO and TZ respectively, have been found, the selection procedure actually selects one solution $\boldsymbol{x}$ randomly from $\{1^n, 0^n\}$ first, each of which with probability $\frac{1}{2}$, then selects the other solution randomly from the remaining solutions $P-\{\boldsymbol{x}\}$, each of which with probability $\frac{1}{i}$. Thus, there are two cases: the two solutions 1^n and 0^n are selected; one selected solution is either 1^n or 0^n and the other selected one is from $P - \{1^n, 0^n\}$. The former case happens with probability $\frac{1}{i}$. In this case, these two solutions can generate one new Pareto optimal solution with probability $\frac{n-i}{n-1}$ by one-point crossover in one step. In the latter case, suppose that the selected solution from $P - \{1^n, 0^n\}$ has $k(0 < k < n)$ leading ones, the number of the solutions having more than k leading ones in the current population is $k'(1 \leq k' \leq i - 1)$ and the other selected solution is 1^n, which happens with probability $\frac{1}{2i}$, these two solutions can generate one new Pareto optimal solution with probability $\frac{n-k-k'}{n-1}$ by one-point crossover in one step. Correspondingly, the other selected solution is 0^n which also happens with probability $\frac{1}{2i}$, then one new Pareto optimal solution can be generated with probability $\frac{k-i+k'}{n-1}$ by one-point crossover in one step. Thus, one new Pareto optimal solution can be generated by one-point crossover in one step with probability $\frac{1}{i} \cdot \frac{n-i}{n-1} + (i-1) \cdot \frac{1}{2i} \cdot (\frac{n-k-k'}{n-1} + \frac{k-i+k'}{n-1}) = \frac{(i+1)(n-i)}{2i(n-1)}$. Since the crossover probability is $\frac{1}{2}$, one new Pareto optimal solution can be generated in one step in the i-th phase with probability at least $\frac{(i+1)(n-i)}{4i(n-1)}$. Thus, the expected steps of the i-th phase is at most $\frac{4i(n-1)}{(i+1)(n-i)}$.

Since two offspring solutions need to be evaluated in one reproduction step, the running time of one step is counted as 2. Thus, the expected running time to find the Pareto front of the optimal Pareto set is at most $2 \cdot \sum_{i=1}^{n-1} \frac{4i(n-1)}{(i+1)(n-i)} \in \Theta(n \log n)$. By combining the running time of the initialization procedure in Lemma 1, the expected running time of REMO with $p_c = 0.5$ to find the Pareto front of the optimal Pareto set of the LOTZ problem is $\Theta(n^2)$. ■

Theorem 2 *On the* COCZ *problem, the expected running time of* REMO *with* $p_c = 0.5$ *is* $\Theta(n \log n)$.

Proof. For the COCZ problem, the size of the Pareto front of the optimal Pareto set is $\frac{n}{2} + 1$. Since the population in the evolutionary process contains at most one Pareto optimal solution for each element in the Pareto front of the optimal Pareto set and Lemma 4, we divide the optimization process into $\frac{n}{2} - 1$ phases, where the i-th phase $(1 \leq i \leq \frac{n}{2} - 1)$ corresponds to the process of evolving the population containing $i+1$ Pareto optimal solutions to that containing $i + 2$ Pareto optimal solutions, and then bound the running time of the whole evolutionary process by summing up the running time bound of each phase.

We consider the i-th phase where the population contains $i + 1$ Pareto optimal solutions. In the selection procedure of REMO, since the two solutions 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$, which are optimal for the two objectives of COCZ respectively, have been found, the algorithm actually select one solution $\boldsymbol{x}$ randomly from $\{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$ first, each of which with probability $\frac{1}{2}$, and then select the other solution randomly from the remaining solutions $P - \{\boldsymbol{x}\}$, each of which with probability $\frac{1}{i}$. Thus, there are two cases: the two solutions 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$ are selected; one selected solution is either 1^n or $1^{\frac{n}{2}}0^{\frac{n}{2}}$ and the other selected one is from $P - \{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$. The former case happens with probability $\frac{1}{i}$. In this case, these two solutions can generate one or two new Pareto optimal solutions with probability at least $\frac{\frac{n}{2}-i}{n-1}$ by one-point crossover in one step. In the latter case, suppose that the selected solution from $P - \{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$ has $k(1 \leq k \leq \frac{n}{2} - 1)$ 0's, the number of the solutions having less than k 0's in the current population is $k'(1 \leq k' \leq i - 1)$ and the other selected solution is 1^n, which happens with probability $\frac{1}{2i}$, these two solutions can generate one or two new Pareto optimal solutions with probability at least $\frac{k-k'}{n-1}$ by one-point crossover in one step. Correspondingly, the other selected solution is $1^{\frac{n}{2}}0^{\frac{n}{2}}$ which also happens with probability $\frac{1}{2i}$, then one or two new Pareto optimal solutions can be generated with probability at least $\frac{\frac{n}{2}-k-i+k'}{n-1}$ by one-point crossover in one step. Thus, one or two new Pareto optimal solutions can be generated by one-point crossover in one step with probability at least $\frac{1}{i} \cdot \frac{\frac{n}{2}-i}{n-1} + (i-1) \cdot \frac{1}{2i} \cdot (\frac{k-k'}{n-1} + \frac{\frac{n}{2}-k-i+k'}{n-1}) = \frac{(i+1)(\frac{n}{2}-i)}{2i(n-1)}$. Since the crossover probability is $\frac{1}{2}$, one or two new Pareto optimal solutions can be generated in one step in the i-th phase with probability at least $\frac{(i+1)(\frac{n}{2}-i)}{4i(n-1)}$. Thus, the expected steps of the i-th phase is at most $\frac{4i(n-1)}{(i+1)(\frac{n}{2}-i)}$.

Since the running time of one step is counted as 2, the expected running time to find the Pareto front of the optimal Pareto set is at most $2 \cdot \sum_{i=1}^{\frac{n}{2}-1} \frac{4i(n-1)}{(i+1)(\frac{n}{2}-i)} \in \Theta(n \log n)$. By combining the running time of the initialization procedure in Lemma 2, the expected running time of REMO with $p_c = 0.5$ to find the Pareto front of the optimal Pareto set of the COCZ problem is $\Theta(n \log n)$. ■

5.2 Analysis of REMO with $p_c = 0$

Theorem 3 *On the* LOTZ *problem, the expected running time of* REMO *with* $p_c = 0$ *is* $\Theta(n^3)$.

Proof. For the LOTZ problem, we define that two Pareto optimal solutions are consecutive if they are neighbor in the sequence $1^n, 1^{n-1}0, \ldots, 0^n$. Since one random bit mutation on a Pareto optimal solution can generate only consecutive Pareto optimal solutions and Lemma 3, the population in any step of the evolutionary process is always constructed by a prefix l and a suffix s of

the sequence $1^n, 1^{n-1}0, \ldots, 0^n$. Then, we divide the optimization process into n phases, where the population in the i-th phase $(1 \leq i \leq n)$ contains $i+1$ Pareto optimal solutions.

In the first phase, $|l| = 1$ and $|s| = 1$, where $|*|$ denotes the length of the sequence $*$. The two solutions 1^n and 0^n are selected. For 1^n, the offspring solution generated by one bit mutation can be accepted only if the last 1 bit is mutated. For 0^n, only if the first 0 bit is mutated, the offspring solution can be accepted. Thus, in one step, two new Pareto optimal solutions will be generated simultaneously with probability $\frac{1}{n^2}$; and only one new Pareto optimal solution will be generated with probability $\frac{2(n-1)}{n^2}$.

Then, we consider the i-th phase $(1 < i \leq n-1)$. In the selection procedure, since 1^n and 0^n are optimal for the two objectives of LOTZ respectively, one solution will be randomly selected from $\{1^n, 0^n\}$ and the other solution will be randomly selected from the remaining solutions. Then, there are two cases.

case 1: $\min\{|l|, |s|\} > 1$. In this case, the solutions 1^n and 0^n by one bit mutation will never generate new Pareto optimal solutions, and only the last solution of the prefix l or the first solution of the suffix s can generate one new Pareto optimal solution with probability $\frac{1}{n}$ by one bit mutation. Thus, one new Pareto optimal solution can be generated in one step with probability $\frac{2}{in}$.

case 2: $\min\{|l|, |s|\} = 1$. Suppose that $|l| = 1$. If the selected solution from $\{1^n, 0^n\}$ is 0^n, mutation on 0^n will never generate new Pareto optimal solutions, and one new Pareto optimal solution can be generated by mutation on the other selected solution with probability $\frac{2}{in}$, since either the solution 1^n or the first solution of the suffix s can generate one new Pareto optimal solution with probability $\frac{1}{n}$. If the selected solution from $\{1^n, 0^n\}$ is 1^n, by mutation on 1^n, one new Pareto optimal solution $1^{n-1}0$ can be generated with probability $\frac{1}{n}$, and by mutation on the other selected solution, one new Pareto optimal solution can be generated with probability $\frac{1}{in}$. Thus, in one step, only one new Pareto optimal solution will be generated while $\min\{|l|, |s|\}$ is still 1 with probability $\frac{1}{in} - \frac{1}{2in^2}$; only one new Pareto optimal solution will be generated while $\min\{|l|, |s|\} > 1$ with probability $\frac{1}{2n} + \frac{1}{2in} - \frac{1}{2in^2}$; and two new Pareto optimal solutions will be generated simultaneously while $\min\{|l|, |s|\} > 1$ with probability $\frac{1}{2in^2}$.

From the analysis above, it can be concluded that the expected running time to find the Pareto front of the optimal Pareto set depends only on the size of the current population (i.e., $|l| + |s|$) and whether $\min\{|l|, |s|\}$ is larger than 1. We denote $E(i)(i \geq 1)$ and $E'(i)(i \geq 3)$ as the expected running time to find the Pareto front of the optimal Pareto set from the i-th phase where $\min\{|l|, |s|\} = 1$, and that from the i-th phase where $\min\{|l|, |s|\} > 1$, respectively.

Then, from the above analysis for the transition behavior in the i-th phase $(1 < i \leq n-1)$, we have

$$\forall 3 \leq i < n : E'(i) = \frac{in}{2} + E'(i+1),$$

and for all integers $i \in [2, n-1)$,

$$E(i) = \frac{2in^2}{(3+i)n-1} + \frac{2n-1}{(3+i)n-1}E(i+1) + \frac{(i+1)n-1}{(3+i)n-1}E'(i+1) + \frac{1}{(3+i)n-1}E'(i+2).$$

It trivially holds that $E(n) = E'(n) = 0$, and $E(n-1) = \frac{2(n-1)n^2}{n^2+2n-1}$. Then, it is not difficult to prove that, for all integers $i \in [3, n]$,

$$E(i) \leq E'(i) = \frac{n(n-i)(n+i-1)}{4}.$$

From the analysis of the first phase and $E(2) = \frac{4n^2}{5n-1} + \frac{2n-1}{5n-1}E(3) + \frac{3n-1}{5n-1}E'(3) + \frac{1}{5n-1}E'(4)$, we have

$$E(1) = \frac{n^2}{2n-1} + \frac{1}{2n-1}E'(3) + \frac{2n-2}{2n-1}E(2)$$
$$= \frac{(13n-9)n^2}{(2n-1)(5n-1)} + \frac{2n-2}{5n-1}E(3) + \frac{6n^2-3n+1}{(2n-1)(5n-1)}E'(3) + \frac{2n-2}{(2n-1)(5n-1)}E'(4).$$

Thus, we have

$$\frac{3}{5}E'(3) < E(1) \leq \frac{(13n-9)n^2}{(2n-1)(5n-1)} + E'(3).$$

Since $E'(3) = \frac{n(n-3)(n+2)}{4} \in \Theta(n^3)$, the expected steps after the initialization procedure to generate the Pareto front of the optimal Pareto set (i.e., $E(1)$) is $\Theta(n^3)$. Since the running time of one step is counted as 2, by combining the running time of the initialization procedure in Lemma 1, the expected running time of REMO with $p_c = 0$ to find the Pareto front of the optimal Pareto set of the LOTZ problem is $\Theta(n^3)$. ■

Theorem 4 *On the COCZ problem, the expected running time of REMO with $p_c = 0$ is $\Theta(n^2)$.*

Proof. For the COCZ problem, we first define that two Pareto optimal solutions $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are consecutive if $|\|\boldsymbol{x}_1\| - \|\boldsymbol{x}_2\|| = 1$, where $\|*\|$ is the 1-norm. That is, the difference of the number of 0's for two consecutive Pareto optimal solutions is 1. Then, we define that a set S of Pareto optimal solutions is consecutive if it contains exactly one solution with i 0's for any $i \in [i_l, i_u]$, where $i_l = \min\{n - \|\boldsymbol{x}\| \mid \boldsymbol{x} \in S\}$ and $i_u = \max\{n - \|\boldsymbol{x}\| \mid \boldsymbol{x} \in S\}$. The solution with i_l 0's and that with i_u 0's in the consecutive set are called l-boundary solution and u-boundary solution, respectively. Since one random bit mutation on a Pareto optimal solution can only generate consecutive Pareto optimal solutions and Lemma 4, the population in the evolutionary process is always constructed by a consecutive set containing 1^n and another consecutive set containing $1^{\frac{n}{2}}0^{\frac{n}{2}}$. Denote l as the consecutive set containing 1^n and s as the consecutive set containing $1^{\frac{n}{2}}0^{\frac{n}{2}}$. Then, we divide the optimization process into $\frac{n}{2}$ phases, where the population in the i-th phase $(1 \leq i \leq \frac{n}{2})$ contains $i+1$ Pareto optimal solutions.

In the first phase, $|l| = 1$ and $|s| = 1$, where $|*|$ denotes the size of the set $*$. The two solutions 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$ are selected. For 1^n, the offspring solution generated by one bit mutation will be accepted if the 1 bit in the second half is mutated. For $1^{\frac{n}{2}}0^{\frac{n}{2}}$, if the 0 bit is mutated, the offspring solution will be accepted. Thus, in one step, two new Pareto optimal solutions will be generated simultaneously with probability $\frac{1}{4}$; and only one new Pareto optimal solution will be generated with probability $\frac{1}{2}$.

Then, we consider the i-th phase $(1 < i \leq \frac{n}{2} - 1)$. In the selection procedure, since 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$ are optimal for the two objectives of COCZ respectively, one solution will be randomly selected from $\{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$ and the other solution will be randomly selected from the remaining solutions. Then, there are two cases.

case 1: $\min\{|l|, |s|\} > 1$. In this case, the solutions 1^n and $1^{\frac{n}{2}}0^{\frac{n}{2}}$ by one bit mutation will never generate new Pareto optimal solutions, and only the u-boundary solution in the set l and the l-boundary solution in the set s can generate one new Pareto optimal solution with probability $\frac{\frac{n}{2}-(|l|-1)}{n}$ and $\frac{\frac{n}{2}-(|s|-1)}{n}$ respectively by one bit mutation. Thus, one new Pareto optimal solution can be

generated in one step with probability $\frac{\frac{n}{2}-(|l|-1)}{in}+\frac{\frac{n}{2}-(|s|-1)}{in}=\frac{n-i+1}{in}$.

case 2: $\min\{|l|,|s|\}=1$. Suppose that $|l|=1$. If the selected solution from $\{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$ is $1^{\frac{n}{2}}0^{\frac{n}{2}}$, mutation on $1^{\frac{n}{2}}0^{\frac{n}{2}}$ will never generate new Pareto optimal solutions, and one new Pareto optimal solution can be generated by mutation on the other selected solution with probability $\frac{1}{2i}+\frac{\frac{n}{2}-(i-1)}{in}$, since 1^n and the l-boundary solution of s can generate one new Pareto optimal solution with probability $\frac{1}{2}$ and $\frac{\frac{n}{2}-(i-1)}{n}$, respectively. If the selected solution from $\{1^n, 1^{\frac{n}{2}}0^{\frac{n}{2}}\}$ is 1^n, by mutation on 1^n, one new Pareto optimal solution can be generated with probability $\frac{1}{2}$, and by mutation on the other selected solution, one new Pareto optimal solution can be generated with probability $\frac{\frac{n}{2}-(i-1)}{in}$. Thus, in one step, only one new Pareto optimal solution will be generated while $\min\{|l|,|s|\}$ is still 1 with probability $\frac{3(\frac{n}{2}-i+1)}{4in}$; only one new Pareto optimal solution will be generated while $\min\{|l|,|s|\}>1$ with probability $\frac{(i+\frac{1}{2})n+i-1}{4in}$; and two new Pareto optimal solutions will be generated simultaneously while $\min\{|l|,|s|\}>1$ with probability $\frac{\frac{n}{2}-i+1}{4in}$.

From the analysis above, it can be concluded that the expected running time to find the Pareto front of the optimal Pareto set depends only on the size of the current population (i.e., $|l|+|s|$) and whether $\min\{|l|,|s|\}$ is larger than 1. We denote $E(i)(i\geq 1)$ and $E'(i)(i\geq 3)$ as the expected running time to find the Pareto front of the optimal Pareto set from the i-th phase where $\min\{|l|,|s|\}=1$, and that from the i-th phase where $\min\{|l|,|s|\}>1$, respectively.

Then, from the above analysis for the transition behavior in the i-th phase $(1<i\leq\frac{n}{2}-1)$, we have, for all integers $i\in[3,\frac{n}{2})$,

$$E'(i)=\frac{in}{n-i+1}+E'(i+1),$$

and for all integers $i\in[2,\frac{n}{2}-1)$,

$$E(i)=\frac{4in}{(i+\frac{5}{2})n-3i+3}+\frac{3(\frac{n}{2}-i+1)}{(i+\frac{5}{2})n-3i+3}E(i+1)$$
$$+\frac{(i+\frac{1}{2})n+i-1}{(i+\frac{5}{2})n-3i+3}E'(i+1)+\frac{\frac{n}{2}-i+1}{(i+\frac{5}{2})n-3i+3}E'(i+2).$$

It trivially holds that $E(\frac{n}{2})=E'(\frac{n}{2})=0$ and $E(\frac{n}{2}-1)=\frac{4n^2-8n}{n^2+12}$. Then it is not difficult to prove that, for all integers $i\in[3,\frac{n}{2}]$,

$$E(i)\leq E'(i)=\sum_{k=i}^{\frac{n}{2}-1}\frac{kn}{n-k+1}.$$

From the analysis of the first phase, we have

$$E(1)=\frac{4}{3}+\frac{1}{3}E'(3)+\frac{2}{3}E(2).$$

Since $E(2)\leq\frac{16n}{9n-6}+E'(3)$, we have

$$\frac{1}{3}E'(3)<E(1)\leq\frac{68n-24}{27n-18}+E'(3).$$

Since $E'(3)=\sum_{k=3}^{\frac{n}{2}-1}\frac{kn}{n-k+1}\in\Theta(n^2)$, the expected steps after the initialization procedure to generate the Pareto front of the optimal Pareto set (i.e., $E(1)$) is $\Theta(n^2)$. Since the running time of one step is counted as 2, by combining the running time of the initialization procedure in Lemma 2, the expected running time of REMO with $p_c=0$ to find the Pareto front of the optimal Pareto set of the COCZ problem is $\Theta(n^2)$. ■

From the comparison between the performance of REMO with $p_c=0$ and that with $p_c=0.5$ on these two problems, it can be concluded that recombination is crucial for the efficiency of REMO on the two problems.

In the above theorems, the running time is expressed in the expectations. They can also be derived in the form of large probabilities, i.e., REMO with $p_c=0.5$ solves the LOTZ problem in $O(n^2)$ time with probability $1-O(1)$, and solves the COCZ problem in $O(n\log n)$ time with probability $1-O(1)$. Compared with other previously analyzed MOEAs, on the LOTZ problem, REMO has the best performance; on the COCZ problem, REMO has better running time than (1+1)-EMO and GEMO, and has better upper bound of running time than SEMO and FEMO. Furthermore, we notice that, since the LOTZ problem was generalized from the LeadingOnes problem and the COCZ problem was generalized from the OneMax problem, the LOTZ and COCZ problems involve single objectives of LeadingOnes and OneMax, respectively. Being aware that (1+1)-EA solves LeadingOnes problem in $\Theta(n^2)$ time [6] and OneMax problem in $\Theta(n\log n)$ time [6], we hypothesize that REMO reaches the lower bounds of running time of any MOEAs on the LOTZ and COCZ problems.

6. DISCUSSIONS AND CONCLUSIONS

Multi-objective evolutionary algorithms have been successfully applied in many practical situations, and some theoretical results of MOEAs have been derived in recent years. However, previous analyzed MOEAs rarely incorporate recombination operators. This paper investigates whether recombination operators can be useful in the scenario of multi-objective optimization.

The Pareto front is a property of MOEA that was not involved in previous theoretical studies of single-objective EAs, thus we investigate whether a recombination operator can have effect on solving the Pareto front of multi-objective problems. We analyze the running time of REMO, a multi-objective evolutionary algorithm with a recombination operator, on the previous studied model problems LOTZ and COCZ. The results show that REMO is more efficient than the previous analyzed MOEAs. We also analyze the running time of REMO turning off recombination on these two problems, and obtain similar performance to that of SEMO. Thus, we conclude that recombination is crucial for the efficiency of REMO on these two problems, which supports that recombination operators can be useful for multi-objective optimization. The analysis of REMO on the two problems discloses that the recombination operator can work by accelerating the filling of the Pareto front through recombining diverse optimal solutions found-so-far. This idea, though proved only in the studied cases, might be hold in general situations and might be useful for designing more efficient MOEAs.

The analysis of REMO on more realistic problems with more kinds of objective functions will be studied. Other ways of incorporating recombination operators in MOEAs are interesting topics that will be investigated in the future.

Acknowledgements

We want to thank the anonymous reviewers for helpful comments and suggestions.

7. REFERENCES

[1] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, Oxford, UK, 1996.

[2] M. Dietzfelbinger, B. Naudts, C. Van Hoyweghen, and I. Wegener. The analysis of a recombinative hill-climber on H-IFF. *IEEE Transactions on Evolutionary Computation*, 7(5):417–423, 2003.

[3] B. Doerr, E. Happ, and C. Klein. Crossover can provably be useful in evolutionary computation. In *Proceedings of the 10th ACM Annual Conference on Genetic and Evolutionary Computation (GECCO'08)*, pages 539–546, Atlanta, GA, 2008.

[4] B. Doerr, D. Johannsen, T. Kötzing, F. Neumann, and M. Theile. More effective crossover operators for the all-pairs shortest path problem. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10)*, pages 184–193, Krakow, Poland, 2010.

[5] B. Doerr and M. Theile. Improved analysis methods for crossover-based algorithms. In *Proceedings of the 11th ACM Annual Conference on Genetic and Evolutionary Computation (GECCO'09)*, pages 247–254, Montreal, Canada, 2009.

[6] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276(1-2):51–81, 2002.

[7] A.-E. Eiben, P.-E. Raué, and Z. Ruttkay. Genetic algorithms with multi-parent recombination. In *Proceedings of the 3rd International Conference on Parallel Problem Solving from Nature (PPSN'94)*, pages 78–87, Jerusalem, Israel, 1994.

[8] S. Fischer and I. Wegener. The one-dimensional Ising model: Mutation versus recombination. *Theoretical Computer Science*, 344(2-3):208–225, 2005.

[9] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. In *Proceedings of the 9th ACM Annual Conference on Genetic and Evolutionary Computation (GECCO'07)*, pages 797–804, London, UK, 2007.

[10] O. Giel. Expected runtimes of a simple multi-objective evolutionary algorithm. In *Proceedings of the 2003 IEEE Congress on Evolutionary Computation (CEC'03)*, pages 1918–1925, Canberra, Australia, 2003.

[11] T. Hanne. On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117(3):553–564, 1999.

[12] T. Hanne. Global multiobjective optimization with evolutionary algorithms: Selection mechanisms and mutation control. In *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization (EMO'01)*, pages 197–212, Zurich, Switzerland, 2001.

[13] J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 127(1):57–85, 2001.

[14] C. Horoba and F. Neumann. Additive approximations of pareto-optimal sets by evolutionary multi-objective algorithms. In *Proceedings of the 10th International Workshop on Foundations of Genetic Algorithms (FOGA'09)*, pages 79–86, Orlando, FL, 2009.

[15] T. Jansen and I. Wegener. The analysis of evolutionary algorithms-a proof that crossover really can help. *Algorithmica*, 34(1):47–66, 2002.

[16] T. Jansen and I. Wegener. Real royal road functions–where crossover provably is essential. *Discrete Applied Mathematics*, 149(1-3):111–125, 2005.

[17] M. Laumanns, L. Thiele, and E. Zitzler. Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions. *IEEE Transactions on Evolutionary Computation*, 8(2):170–182, 2004.

[18] M. Laumanns, L. Thiele, E. Zitzler, E. Welzl, and K. Deb. Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature (PPSN'02)*, pages 44–53, Birmingham, UK, 2002.

[19] P. K. Lehre and X. Yao. Crossover can be constructive when computing unique input output sequences. In *Proceedings of the 7th International Conference on Simulated Evolution and Learning (SEAL'08)*, pages 595–604, Melbourne, Australia, 2008.

[20] G. Lin and X. Yao. Analysing crossover operators by search step size. In *Proceedings of the 1997 IEEE Congress on Evolutionary Computation (CEC'97)*, pages 107–110, Indianapolis, IN, 1997.

[21] T. Murata and H. Ishibuchi. MOGA: Multi-objective genetic algorithms. In *Proceedings of the 1995 IEEE Congress on Evolutionary Computation (CEC'95)*, pages 289–294, Perth, Australia, 1995.

[22] F. Neumann. Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. *European Journal of Operational Research*, 181(3):1620–1629, 2007.

[23] F. Neumann and J. Reichel. Approximating minimum multicuts by evolutionary multi-objective algorithms. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN'08)*, pages 72–81, Dortmund, Germany, 2008.

[24] F. Neumann and M. Theile. How crossover speeds up evolutionary algorithms for the multi-criteria all-pairs-shortest-path problem. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10)*, pages 667–676, Krakow, Poland, 2010.

[25] F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 5(3):305–319, 2006.

[26] P. Oliveto, J. He, and X. Yao. Analysis of population-based evolutionary algorithms for the vertex cover problem. In *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (CEC'08)*, pages 1563–1570, Hong Kong, China, 2008.

[27] J. Richter, A. Wright, and J. Paxton. Ignoble trails-where crossover is provably harmful. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN'08)*, pages 92–101, Dortmund, Germany, 2008.

[28] O. Rudenko and M. Schoenauer. Dominance based crossover operator for evolutionary multi-objective algorithms. In *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN'04)*, pages 812–821, Birmingham, UK, 2004.

[29] G. Rudolph. Evolutionary search for minimal elements in partially ordered sets. In *Proceedings of the 7th International Conference on Evolutionary Programming (EP'98)*, pages 345–353, San Diego, CA, 1998.

[30] G. Rudolph. On a multi-objective evolutionary algorithm and its convergence to the pareto set. In *Proceedings of the 1998 IEEE Congress on Evolutionary Computation (CEC'98)*, pages 511–516, Piscataway, NJ, 1998.

[31] G. Rudolph and A. Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *Proceedings of the 2000 IEEE Congress on Evolutionary Computation (CEC'00)*, pages 1010–1016, Piscataway, NJ, 2000.

[32] W. Spears. *Evolutionary Algorithms: The Role of Mutation and Recombination*. Springer Verlag, Berlin, Germany, 2000.

[33] D. Sudholt. Crossover is provably essential for the ising model on trees. In *Proceedings of the 7th ACM Annual Conference on Genetic and Evolutionary Computation (GECCO'05)*, pages 1161–1167, Washington DC, 2005.

[34] R. A. Watson. Analysis of recombinative algorithms on a non-separable buildingblock problem. In *Proceedings of the 6th International Workshop on Foundations of Genetic Algorithms (FOGA'00)*, pages 69–89. San Mateo, CA, 2000.

[35] Y. Yu, C. Qian, and Z.-H. Zhou. Towards analyzing recombination operators in evolutionary search. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10)*, pages 144–153, Krakow, Poland, 2010.

[36] Y. Yu and Z.-H. Zhou. A new approach to estimating the expected first hitting time of evolutionary algorithms. *Artificial Intelligence*, 172(15):1809–1832, 2008.