A new exact method and matheuristics for bi-objective 0/1 ILPs: Application to FTTx-network design

Markus Leitner*[1], Ivana Ljubić†[1], Markus Sinnl‡[1], and Axel Werner§[2]

[1]Department of Statistics and Operations Research, Faculty of Business, Economics and Statistics, University of Vienna, Austria
[2]Zuse Institute Berlin, Berlin, Germany

Abstract

Heuristics and metaheuristics are inevitable ingredients of most of the general purpose MIP solvers today, because of their contribution to the significant boost of the performance of exact methods. In the field of bi/multi-objective optimization, the interaction between the exact and metaheuristic communities is still fairly low. This article is one of the first steps towards reducing this gap and bringing the attention of both communities to still unexplored possibilities for performance improvements of exact and heuristic multi-objective optimization algorithms.
We focus on bi-objective optimization problems whose feasible solutions can be described as 0/1 integer linear programs and propose a new exact method called *adaptive search in objective space* (ASOS). ASOS combines features of the ϵ-constraint method with the binary search in the objective space. In addition, two matheuristics, *boundary induced neighborhood search* (BINS) and *directional local branching* are proposed. Their main idea is to combine the features and explore the neighborhoods of solutions that are relatively close in the objective space. Finally, a *two-phase ILP-based heuristic framework* relying on BINS and directional local branching is proposed.
Our new methods are computationally evaluated on two problems of particular relevance for the design of FTTx-networks. Comparison with other known exact methods (relying on the exploration of the objective space) is conducted on a set of realistic benchmark instances representing telecommunication access networks from Germany.

1 Introduction

Recent advances in the development of general purpose mixed integer programming (MIP) solvers have led to an increased popularity of exact MIP-based approaches for bi/multi-objective optimization. Two main research directions can be observed: branch-and-bound based algorithms (performing the search in the decision space, see, e.g., (1; 2; 3; 4)), and iterative exact methods (performing the search in the objective space, see, e.g., (5; 6; 7)). A large body of work is available in the field of meta-heuristics as well (see, e.g., (8; 9)). Not much has been done, however, in the development of *matheuristics* for bi-objective optimization. After many decades of independent research in mixed integer programming and metaheuristics for single-objective (combinatorial) optimization, researchers came upon realization that significant advantages can be drawn from synergetic effects of their hybridization. Nowadays, most of the general purpose MIP solvers contain (meta)heuristics as their inevitable features that also significantly contribute to the boost of their

*markus.leitner@univie.ac.at
†ivana.ljubic@univie.ac.at
‡markus.sinnl@univie.ac.at
§werner@zib.de

performance (see, e.g., (10; 11; 12)). In the field of bi/multi-objective optimization, this is still not the case, and the interaction between the communities is still fairly low. This article is one of the first steps towards reducing this gap and bringing the attention of both communities to still unexplored possibilities for performance improvements of exact and heuristic multi-objective optimization methods.

In this article we consider bi-objective combinatorial optimization problems that can be modeled as bi-objective 0/1 integer linear programs (ILPs). Our contribution is twofold:

1. We propose a new exact ILP-based method, *adaptive search in objective space* (ASOS) that explores the objective space in order to establish the complete Pareto front. This exact solution framework is based on combining the *binary search in objective space (BSOS)* (7; 13)) and the *ϵ-constraint method* (5; 14). Our framework is guided by (the absence of) heuristic solutions with the main goal to benefit from the advantages of the two methods while avoiding their individual drawbacks.

2. We propose two matheuristics for bi-objective 0/1 ILPs: *boundary induced neighborhood search* (BINS) and *directional local branching*, that are bi-objective counterparts of two efficient matheuristics for single-objective optimization, relaxation induced neighborhood search (RINS) (10) and local branching (11), respectively. The two matheuristcs are then embedded into an *two-phase ILP-based heuristic* that is used to approximate the Pareto front for large instances.

The development of these new methods is motivated by our computational experience with certain bi-objective problems arising in the design of FTTx-networks, showing that established iterative exact methods are not able to discover the complete Pareto front for most of the instances relevant for these practical applications.

Planning of Telecommunication Access Networks One main step in cost-efficient planning of telecommunication access networks is to find an (optimal) assignment of potential customers to different available *technologies* (*architectures*), i.e., a *deployment strategy.* Commonly used architectures include fiber-to-the-air (FTTA), fiber-to-the-curb (FTTC), fiber-to-the-building (FTTB), and fiber-to-the-home (FTTH). Network providers are faced with a natural question: which customers to serve with which technology so as to minimize the total investment costs while maximizing the quality of service. It is immediate that optimal deployment decisions are naturally subject to multiple objectives. Designing optimal FTTH networks is typically modeled as a variant of the *Steiner tree problem (STP)* in graphs (see, e.g., (15; 16)) while variants of the *Connected Facility Location Problem (ConFL)* have been used for planning FTTC networks, cf. (17; 18). We introduce the *multi-objective k-architecture connected facility location problem* (MOkA-ConFL), generalizing connected facility location to more than two architectures and to multiple-objectives. The computational success of our new approaches is demonstrated on bi-objective problems, that arise as special cases of MOkAConFL with practical applications. These problems are the *bi-objective connected facility location problem* (BOConFL) and the *bi-objective two-architecture connected facility location problem* (BOTAConFL).

Outline of the Article Required concepts from bi-objective optimization and necessary notation are summarized in the remainder of this section. Based on a short review of the BSOS and the ϵ-constraint method, we detail our new method, adaptive search in objective space, in Section 2. Section 3 introduces our general-purpose ILP-heuristics for the bi-objective case and discusses the new heuristic framework while Section 4 introduces MOkAConFL, its bi-objective variants that will be used in our computational study, and details necessary for adaptating our frameworks to these particular problems. Further implementation details and the results of our computational study on the considered benchmark problems are summarized in Section 5. Finally, in Section 6, conclusions and possible directions for future research are provided.

Basic Definitions and Notation Next, we introduce necessary notation and recall some basic terminology for bi-objective optimization, see, e.g., (19) for a more detailed overview. Throughout this article, we will only consider problems in minimization form and will assume that all input data is integral. For a bi-objective optimization problem $\min_{\sigma \in \mathcal{P}}(z_1(\sigma), z_2(\sigma))$, its feasible region $\mathcal{P}$ is called *decision space* and $Z = \{(z_1(\sigma), z_2(\sigma)) : \sigma \in \mathcal{P}\}$ is the set of images of the points in $\mathcal{P}$ in the *objective space* $\mathbb{R}^2$.

For ease of notation, for $\sigma^i \in \mathcal{P}$, let $z_1^i = z_1(\sigma^i), z_2^i = z_2(\sigma^i)$ and $z^i = (z_1^i, z_2^i)$. Moreover, we will also sometimes slightly abuse notation, and use z^i (i.e., a point in the objective space) to also refer to a solution σ^i (i.e., a point in the decision space) with $z_1(\sigma^i) = z_1^i$, $z_2(\sigma^i) = z_2^i$. This is only done when it is clear from the context, that such a solution exists.

A solution $\sigma^* \in \mathcal{P}$ is called *Pareto optimal (efficient)*, if and only if there is no solution $\sigma' \in \mathcal{P}$ such that $z_i(\sigma') \leq z_i(\sigma^*)$, $i = 1, 2$, with at least one strict inequality. The objective point $z^* = (z_1(\sigma^*), z_2(\sigma^*))$ corresponding to an efficient solution σ^* is called *non-dominated.* The set of all Pareto optimal solutions is denoted by P_E and the set of all non-dominated points, also called *Pareto front* or non-dominated frontier, by $\mathcal{Z}$. An objective point $z(\bar{\sigma})$ corresponding to a solution $\bar{\sigma}$ is called *weakly dominated* if there exists another solution $\hat{\sigma}$ with $z_i(\hat{\sigma}) \leq z_i(\bar{\sigma})$, $i = 1, 2$ and $z_i(\hat{\sigma}) = z_i(\bar{\sigma})$ for either $i = 1$ or $i = 2$.

The set of efficient solutions can be partitioned into two subsets, those whose objective vectors lie on the convex hull of the Pareto front, which are usually called *supported* efficient solutions, and the remaining, so-called *non-supported* efficient solutions; the points in the objective space are called analogously. The boundary points (z_1^I, z_2^N) and (z_1^N, z_2^I) of the Pareto front that are defined by the *ideal point* $z_i^I = \min\{z_i(\sigma) : \sigma \in \mathcal{P}\}$ and the *nadir point* $z_i^N = \min\{z_i(\sigma) : \sigma \in \mathcal{P}, z_j(\sigma) \leq z_j^I, j \neq i\}$, $i = 1, 2$, play an important role in most iterative solution methods. Given the objective vectors of two solutions σ^a and σ^b with $z_2(\sigma^a) > z_2(\sigma^b)$, we will denote by $[z^a, z^b]$ the *rectangle* $\{(z_1, z_2) \mid z_1^a \leq z_1 \leq z_1^b, z_2^b \leq z_2 \leq z_2^a\}$ in the objective space defined by these two solutions.

When using heuristic methods, or when an exact method cannot terminate due to given memory- or timelimits, one usually ends up with an approximate Pareto front. The quality of such an approximation can be assessed by the *hypervolume indicator*, see, e.g., (20) for an overview on performance assessment methods of multi-objective optimization algorithms. Given a set of solutions $\hat{P}_E = \{\sigma^1, \dots, \sigma^n\}$, in bi-objective minimization, the hypervolume indicator $H(\hat{P}_E)$ is defined as the area dominated by the solutions in $\hat{P}_E$, i.e., the area covered by $\bigcup_{i=1}^n [(z_1^i, z_2^N), (z_1^N, z_2^i)]$. The hypervolume indicator attains a maximum for the complete Pareto front P_E, and it generally provides a lower bound for the area dominated by P_E. A higher value of the hypervolume indicator usually indicates a better approximation of the non-dominated frontier. For exact methods, in this article we introduce the *relaxed hypervolume indicator* rH, which is based on calculating the upper bound of the hypervolume, and in addition to this we also define a *hypervolume gap indicator* (note that a concept of this name is mentioned in (6) without explicit definition) which is calculated using H and rH. Details are given in Section 5.2.

2 New Exact Solution Framework

As mentioned in the introduction, ILP-based exact methods for multi-objective optimization typically follow one of the two patterns: they either rely on the search of the decision space, or they establish the complete Pareto front by exploring the objective space. The methods studied in this article fall into the latter category, and we will refer to them as *iterative methods.* Typically, an iterative scheme is defined, in which specifically constrained ILPs are solved, and the objective space is further explored based on the obtained optimal solutions.

Before introducing the new exact framework in Subsection 2.3, we detail the two iterative approaches it is mainly composed of. Given weights ω_1, ω_2 and bounds ϵ_1, ϵ_2, we will denote by *iteration* the process of solving ILP $P(\omega_1, \omega_2, \epsilon_1, \epsilon_2) = \min\{\omega_1 z_1(\sigma) + \omega_2 z_2(\sigma) : \sigma \in \mathcal{P}, z_1(\sigma) \leq \epsilon_1, z_2(\sigma) \leq \epsilon_2\}$, assuming that "$\sigma \in \mathcal{P}$" symbolically stands for a 0/1 ILP description of the set of feasible solutions. The optimal solution of such an iteration will be denoted by σ^* while δ_1 and δ_2 will denote the greatest common divider (gcd) of all coefficients in $z_1(\sigma)$ and $z_2(\sigma)$, respectively. List *Sol* is the *solution pool.* It stores the current approximation of the Pareto front. *Sol* is initialized with the boundary points (z_1^I, z_2^N) and (z_1^N, z_2^I) of the non-dominated frontier. This process is denoted as *initBP*() in the following. Solutions generated by some heuristic procedure that are Pareto optimal with respect to the current approximation of the Pareto front are called *candidate solutions* or *potentially* Pareto optimal solutions. Note that such solutions will also be stored in

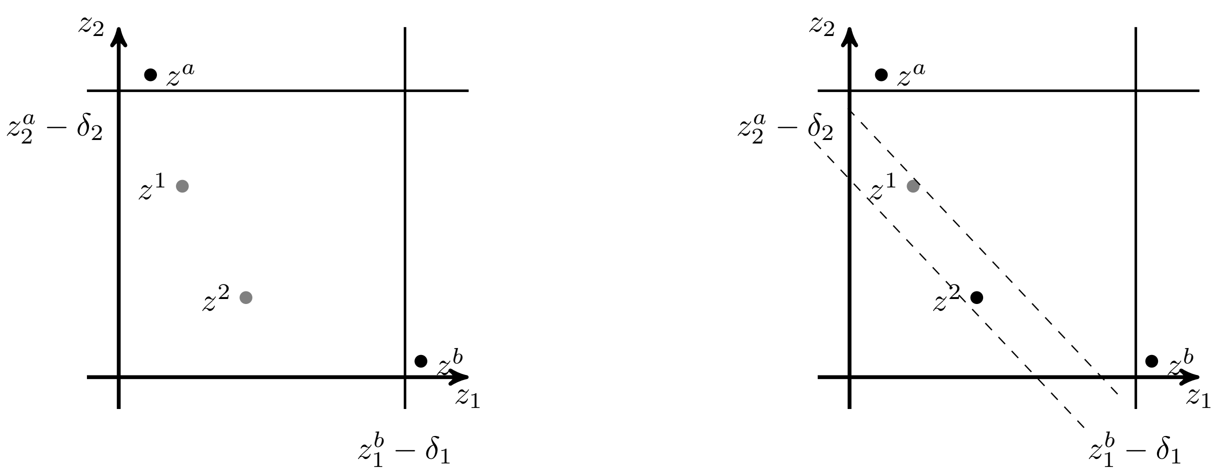
Sol. *Sol* is used to initialize the ILP of an iteration with a starting solution and also to guide our exact framework. More details are given in Subsection 2.3.

2.1 Binary Search in Objective-Space (BSOS)

The *binary search in objective-space* (BSOS) (7; 13) which is summarized in Algorithm 1 is a variant of the weighted-sum approach. In contrast to the standard weighted-sum method for bi-objective optimization (see, e.g., (21)), BSOS is able to discover non-supported efficient solutions as well. To this end, the corner points z^a and z^b defining the rectangle $[z^a, z^b]$ of a current iteration are cut off with additional constraints $z_1(\sigma) \leq z_1^b - \delta_1$ and $z_2(\sigma) \leq z_2^a - \delta_2$, which are added to the weighted-sum optimization problem, i.e., $P(\omega_1, \omega_2, z_1^b - \delta_1, z_2^a - \delta_2)$ is solved, see Figure 1 for an illustration.

Algorithm 1 Binary Search in Objective Space

$Sol \leftarrow initBP()$
$I \leftarrow \{[(z_1^I, z_2^N), (z_1^N, z_2^I)]\}$
while $I \neq \emptyset$ **do**
  select a rectangle $[z^a, z^b] \in I$ and remove it from I
  $\sigma^* \leftarrow \text{argmin } P(\omega_1, \omega_2, z_1^b - \delta_1, z_2^a - \delta_2)$
  if $\sigma^* \neq \emptyset$ **then**
    $Sol \leftarrow Sol \cup \{z(\sigma^*)\}$
    if $z_2^a - z_2(\sigma^*) \geq 2\delta_2$ **then**
      $I \leftarrow I \cup \{[z^a, z(\sigma^*)]\}$
    if $z_1^b - z_1(\sigma^*) \geq 2\delta_1$ **then**
      $I \leftarrow I \cup \{[z(\sigma^*), z^b]\}$



(a) z^1, z^2 are not yet discovered Pareto optimal solutions. The bold lines give the constraints on $z_1(\sigma)$ and $z_2(\sigma)$.

(b) Dashed lines are level lines of the objective function. Non-dominated point z^2 is discovered.

Figure 1: Iteration in the binary search in objective space. Rectangle $[z^a, z^b]$ is explored.

We observe that up to $2|\mathcal{Z}| + 3$ ILPs have to be solved to determine a Pareto front consisting of $|\mathcal{Z}|$ points using the BSOS. Besides the four initial ones to obtain the two boundary points and $|\mathcal{Z}| - 2$ to obtain all remaining points, we also need to solve at most $|\mathcal{Z}| + 1$ additional ILPs, one for each empty interval between two Pareto optimal solutions. The latter fact resembles the main weakness of this methods, since the chosen

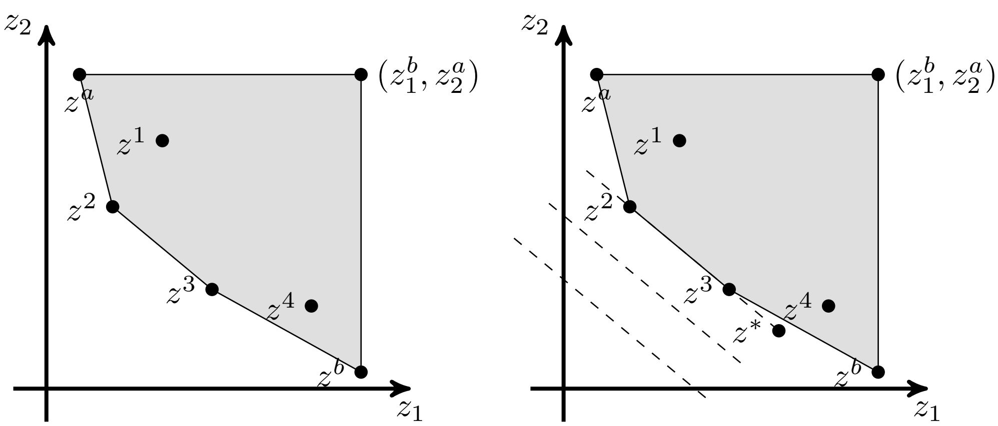
ILP solver has to prove infeasibility for each of these problems, a process that typically takes significantly longer than solving an ILP with at least one feasible solution. The order in which rectangles (stored in the queue I in the pseudo-code given in Algorithm 1) are proceeded, may significantly influence the performance of this approach. In our implementation, we choose rectangles according to their contribution to the relaxed hypervolume rH (see Section 5.2 for further details).

While usually weights $\omega_1 = z_2^a - z_2^b$ and $\omega_2 = z_1^b - z_1^a$ that yield an objective function parallel to the line through $[z^a, z^b]$ are chosen, the method works in principle for any selection of $\omega_1, \omega_2 > 0$. Next, we propose an approach that can possibly prove Pareto optimality of multiple, already known feasible solutions (such solutions could have been found, for example, by heuristics) in a single iteration. Let $\mathcal{P}^C$ be the set of all candidate solutions with images lying in the currently considered rectangle $[z^a, z^b]$, and Z^C be the set of corresponding images. The approach exploits the result of the following Lemma, see Figure 2 for an illustration.

Lemma 1. *Consider a facet $\mathcal{F}$ of the polygon $\operatorname{conv}(z^a, z^b, Z^C, (z_1^b, z_2^a))$, defined by two points $z^i = z(\sigma^i)$ and $z^j = z(\sigma^j)$ with $\sigma^i, \sigma^j \in \mathcal{P}^C$ and $z_2^i > z_2^j$ such that $z^a, z^b \notin \mathcal{F}$ and let $\omega_1 := z_2^i - z_2^j$, $\omega_2 := z_1^j - z_1^i$, and $z^\omega := \omega_1 z_1^i + \omega_2 z_2^i$. Furthermore, let $\mathcal{P}^\omega$ be the set of all candidate solutions $\sigma \in \mathcal{P}^C$ such that $z(\sigma) \in \mathcal{F}$. If the optimal solution value of $P(\omega_1, \omega_2, z_1^b - \delta_1, z_2^a - \delta_2)$ is equal to z^ω, then all solutions in $\mathcal{P}^\omega$ are Pareto optimal.*

Proof. The obtained solution (with objective value z^ω) is Pareto optimal by the validity of the BSOS method (since $\omega_1, \omega_2 > 0$). Since all solutions in P^ω have objective points on $\mathcal{F}$ with the same objective value (by the choice of ω_1, ω_2), they are all Pareto optimal. □

Figure 2b illustrates the result of Lemma 1 and also shows that more than two new rectangles may be obtained when using this approach.



(a) $conv(z^a, z^b, Z^C, (z_1^b, z_2^a))$.

(b) Non-dominated point z^* with objective value z^ω is found.

Figure 2: $Z^C = \{z^1, z^2, z^3, z^4\}$. The weights are chosen based on z^2 and z^3 and the dashed lines are level lines of the resulting objective function. The non-dominated point z^* allows us to conclude that z^2 and z^3 are also non-dominated. The new rectangles are $[z^a, z^2]$, $[z^2, z^3]$, $[z^3, z^*]$ and $[z^*, z^b]$.

2.2 ϵ-Constraint Method

In the ϵ-constraint method, which is one of the most popular methods for solving bi-objective combinatorial optimization problems (see, e.g., (5) for a recent application), one of the two objectives, say z_2, is transformed into a constraint, i.e., problem $P(1, 0, \infty, \epsilon)$ is considered. By systematically decreasing parameter ϵ from z_2^N to z_2^I the Pareto front is determined.

In this basic version, weakly dominated points may be found. These points can simply be removed in a post-processing phase. Moreover, this problem can also be resolved by using lexicographic minimization, i.e., by either solving $P(1, \gamma, \infty, \epsilon)$ for an appropriately small value of γ or by solving a second ILP $\min\{z_2(\sigma) : \sigma \in \mathcal{P}, z_1(\sigma) = z_1(\sigma^*)\}$, in each iteration. According to our computational experience it is much faster to simply remove weakly dominated points in the end, rather than using lexicographic minimization. A drawback of the method arises from the fact that it does not generate a good approximation of the Pareto front early (since it searches the objective space from top left to bottom right). Consequently, the hypervolume typically increases only slowly during the solution process.

2.3 Adaptive Search in Objective Space (ASOS)

The motivation of our new exact solution framework, *adaptive search in objective space* (ASOS) (see Algorithm 2) is to combine the ϵ-constraint method and BSOS in such a way that we benefit from their advantages and do not face their drawbacks. The default method is BSOS, since it quickly computes an approximation of the Pareto frontier and does not return weakly Pareto optimal solutions. To avoid proving infeasibility of ILPs associated to some rectangle $[z^a, z^b]$, we aim to efficiently guess when such a case might occur and call the ϵ-constraint method with $P(1, 0, \infty, z_2^a - \delta_2)$ instead. If our prediction was correct the ϵ-constrained method will return a solution z^c with $z_2^c = z_2^b$ and doing so is typically much faster than proving emptiness of the interval by BSOS. If, on the contrary, a new solution is found by the ϵ-constraint method, a new Pareto optimal solution σ^* is derived using its lexicographic variant. Subsequently, the rectangle $[z(\sigma^*), z^b]$ is added to the queue of unprocessed rectangles. Note that, in that case, the ϵ-constraint method implicitly proves that the rectangle $[z_a, z(\sigma^*)]$ does not contain further non-dominated points.

ASOS uses the set of so-far discovered and not yet dominated solutions (the solution pool *Sol*) to decide which solution method to apply for a given rectangle $[z^a, z^b]$ as follows: If *Sol* does not contain any solution lying in the rectangle $[z^a, z^b]$, we conclude that it is likely that no such solution exists and apply the ϵ-constraint method (with σ^b as starting solution). On the contrary, if at least one solution in $[z^a, z^b]$ has been found previously, BSOS is applied. We use the most promising solution (i.e., the one with minimum objective value) from the solution pool as starting solution. The process of passing this starting solution as initial incumbent to the ILP solver is denoted by *setStartingSolution* in Algorithm 2.

Clearly, our framework relies on the effective population of the solution pool. Besides adding all incumbent solutions found throughout previous iterations (which might turn out to be Pareto optimal in subsequent iterations), we propose to use the general purpose ILP-based heuristic, *boundary induced neighborhood search* (BINS), see next section. Observe that in the first ten iterations of ASOS we run binary search to collect diverse solutions for *Sol*.

Our Algorithm also uses further generic acceleration methods denoted with *updateBranchingPriorities*(σ^*) and *updateConstraintPool*$()$, which are discussed below. Necessary adaptations of these generic ideas to the considered problems are described in Section 5.1. These acceleration methods and the solution pool are used in all methods considered in our computational study. We also experimented with *visit* and *cover inequalities*, cf. (22), but they did not give promising results in preliminary tests.

Constraint Pool Many combinatorial optimization problems can be modeled as ILPs with a huge (potentially exponential) number of constraints that are dynamically added using branch-and-cut. To this end, an appropriate oracle (separation method) is called which identifies and adds (separates) violated constraints. In iterative solution methods some of these inequalities will likely be added in several iterations. Thus, a constraint pool (see, (7; 22)) stores them and is checked for violated constraints before calling the computationally more expensive separation routine. In our default setting, the pool is re-initialized with each new iteration, and only constraints violated in the previous iteration are kept in the pool.

Algorithm 2 Adaptive Search in Objective Space (ASOS)

$Sol \leftarrow initBP()$
$iterations \leftarrow 0$
$I \leftarrow \{[(z_1^I, z_2^N), (z_1^N, z_2^I)]\}$
while $I \neq \emptyset$ **do**
  $iterations \leftarrow iterations + 1$
  select a rectangle $[z^a, z^b] \in I$ and remove it from I
  if $\exists \sigma \in Sol : z(\sigma) \in [z^a, z^b] \ \vee \ iterations \leq 10$ **then**
    $setStartingSolution(\sigma^a, \sigma^b)$
    $\sigma^* \leftarrow \text{argmin } P(\omega_1, \omega_2, z_1^a - \delta_1, z_2^b - \delta_2)$
    if $\sigma^* \neq \emptyset$ **then**
      $updateBranchingPriorities(\sigma^*)$, $updateConstraintPool()$
      $Sol \leftarrow Sol \cup \{z(\sigma^*)\}$
      if $z_2^a - z_2(\sigma^*) \geq 2\delta_2$ **then**
        $I \leftarrow I \cup \{[z^a, z(\sigma^*)]\}$
        $Sol \leftarrow Sol \cup BINS(z^a, z(\sigma^*))$
      if $z_1^b - z_1(\sigma^*) \geq 2\delta_1$ **then**
        $I \leftarrow I \cup \{[z(\sigma^*), z^b]\}$
        $Sol \leftarrow Sol \cup BINS(z(\sigma^*), z^b)$
  else
    $setStartingSolution(\sigma^b)$
    $\sigma^* \leftarrow \text{argmin } P(1, 0, \infty, z_2^a - \delta)$
    if $z_2(\sigma^*) \neq z_2^b$ **then**
      $updateBranchingPriorities(\sigma^*)$, $updateConstraintPool()$
      $\sigma^* \leftarrow \text{argmin } \{z_1(\sigma) : \sigma \in \mathcal{P}, z_2(\sigma) = z_2(\sigma^*)\}$
      $updateBranchingPriorities(\sigma^*)$, $updateConstraintPool()$
      $Sol \leftarrow Sol \cup \{z(\sigma^*)\}$
      if $z_2(\sigma^*) - z_2^b \geq 2\delta_2$ **then**
        $I \leftarrow I \cup \{[z(\sigma^*), z^b]\}$
        $Sol \leftarrow Sol \cup BINS(z(\sigma^*), z^b)$

Adaptive Branching Iterative solution frameworks for bi-objective problems allow to better guide the branching decisions of the current ILP iteration by exploiting knowledge gained during previous iterations (22). One natural way that we make use of, is to increase the branching priority of a binary variable each time the corresponding object is included in a Pareto optimal solution. We refer to this branching strategy as *adaptive branching*.

3 ILP-Based Heuristics for Bi-objective Integer Programming

In this section, we propose a new, generic two-phase heuristic framework based on black-box ILP procedures that aims to overcome the following two severe drawbacks of iterative ILP-based exact methods: a single ILP iteration may (i) require too much time or (ii) run out of memory. In both cases, only a very small part of the Pareto front may be discovered and iterative methods may not be able to continue in a reasonable way since they usually rely on the identification of Pareto optimal solutions of previous iterations. Our framework, which will be detailed in Section 3.3, is based on *boundary solution induced neighborhood search* (BINS) and *directional local branching*, cf. Sections 3.1 and 3.2. The latter two are new multi-objective generalizations of well-established single-objective black-box ILP heuristics, namely *RINS* (10) and *local branching* (11), respectively. In the following, we describe our methods for 0/1 ILPs, but we also point out that our methods can be easily adapted to general ILPs.

3.1 Boundary Solution Induced Neighborhood Search (BINS)

When LP-relaxations are solved within a branch-and-bound procedure for ILPs, some of the integer decision variables may be (almost) integer in an optimal LP-solution, while others are not. To produce high-quality feasible solutions, variable fixing heuristics try to fix decision variables in an intelligent way by using information gained during the solution process. In case of the *relaxation induced neighborhood search* (RINS) (10), the value of the LP-relaxation is used to fix the decision variables. Inspired by these ideas, BINS aims to exploit the fact that Pareto optimal solutions corresponding to non-dominated points in some rectangle $[z^a, z^b]$ often share solution characteristics with the boundary solutions σ^a and σ^b. Let Σ be the set of indices of variables of the considered problem, $F_0 = \{i \in \Sigma : \sigma_i^a = \sigma_i^b = 0\}$ and $F_1 = \{i \in \Sigma : \sigma_i^a = \sigma_i^b = 1\}$ be the sets of variables that are equal to zero and one, respectively, in both solutions. We fix (some of) the variables whose values are identical in these boundary solutions in order to find a new potentially Pareto optimal solution by solving $P(\omega_1, \omega_2, z_1^a - \delta_1, z_2^b - \delta_2)$ extended by constraints $\sigma_i = 0$, for all $i \in F_0$ and $\sigma_i = 1$, for all $i \in F_1$. Since in that case we are solving a restricted variant of a BSOS iteration with a potentially large number of variables fixed to zero or one, one can expect to find feasible solutions extremely fast. As in RINS, one may fix only variables from F_0 (F_1) to zero (one) or impose both constraints. Note that the efficiency of BINS clearly depends on the size of the rectangle, the number of feasible solutions inside, and typically benefits from increasing sizes of sets F_0 and F_1.

3.2 Directional Local Branching

Local branching (11) is a generic ILP-based method to effectively search a neighborhood of a feasible reference solution $\bar{\sigma}$. Given parameter $n \in \mathbb{N}$ and the set of indices $S^1 = \{i \in \Sigma : \bar{\sigma}_i = 1\}$ of variables whose values are equal to one in $\bar{\sigma}$, a neighborhood $N(n, \bar{\sigma})$ of size n for this reference solution $\bar{\sigma}$ is constructed by adding the following local branching constraint to the problem formulation.

$$\sum_{i \in S^1} (1 - \sigma_i) + \sum_{i \in \Sigma \setminus S^1} \sigma_i \leq n. \tag{1}$$

Constraint (1) ensures that at most n variables of a feasible solution attain values different to the value of the reference solution $\bar{\sigma}$. Since the associated feasible set is typically small, solving an ILP with the local branching constraint (and using the reference solution as initial solution) often allows to derive an improved solution extremely fast.

Directional local branching generalizes local branching to bi-objective problems as follows: Given a (potentially Pareto optimal) solution σ^P we aim to identify yet unknown (potentially Pareto optimal) solutions close to σ^P. To this end we minimize one of the objective functions (i.e., z_i, $i = 1, 2$) while restricting the feasible space to a neighborhood of σ^P. More precisely, we add a local branching constraint for σ^P, minimize in z_i-direction and additionally add an ϵ-constraint on objective z_j, $j \neq i$, to avoid computing a solution with the ideal-point value for objective i. For example, for $i = 1, j = 2$, we solve the problem $P(1, 0, \infty, \epsilon)$ with an additional constraint as defined in equation (1). Two variants for choosing the right-hand-side of this ϵ-constraint can be considered: (i) z_j^P or (ii) $z_j^P - \delta_j$, $j \neq i$. The latter variant explicitly ensures that a solution different to σ^P is produced (if a suitable one exists in the given neighborhood).

Observe that our new approach offers a generic way to perform *multi-directional local search* (23). By letting the ILP-solver explore the neighborhood, our approach avoids the implementation of (possibly complicated) problem-dependent local-search procedures whose execution may also be time-consuming. The idea of the directional local branching can be straight-forwardly adapted to more than two objectives: one objective is part of the minimization function, the remaining objectives are bounded from above, and the neighborhood constraint is added.

3.3 The Two-Phase ILP-based Heuristic Framework

In this section we describe a new ILP-based heuristic framework whose main ingredients are BINS and directional branching described above. Inspired by the two-phase methods (TPM) for bi-objective combinatorial optimization problems (see, e.g., (21) or (24)) our ILP-based heuristic framework consists of two phases. Using a weighted-sum approach, the first phase aims to discover the set of all supported non-dominated points, which likely provides a good approximation of the Pareto front. A timelimit t_{FP} is applied in each iteration, to avoid potentially arising excessive runtimes of single iterations. In addition, we apply BINS to each rectangle $[z^a, z^b]$ identified in this first phase in order to find further (potentially Pareto optimal) solutions and populate the solution pool *Sol*.

Starting off with the set of solutions found in the first phase, the second phase iteratively refines the approximate Pareto front by applying directional local branching as long as improved (i.e., non-dominated) solutions can be found. The framework is summarized in Algorithm 3. As above, *Sol* contains the set of currently non-dominated points. The set *newSol* used in the second phase initially contains all points discovered in the first phase. The neighborhood of each single point in *newSol* is explored and the newly discovered non-dominated points, which are temporary stored in the set *neighbors*, are passed over to the next iteration i.e., *newSol* is reset to non-dominated solutions from *neighbors*.

Algorithm 3 Two-Phase ILP-based Heuristic Framework

```
Sol ← initBP()
I ← {[(z_1^I, z_2^N), (z_1^N, z_2^I)]}
while I ≠ ∅ do
  select a rectangle [z^a, z^b] ∈ I and remove it from I
  setStartingSolution(σ^a, σ^b)
  σ* ← argmin P(ω_1, ω_2, ∞, ∞)
  if σ* ≠ ∅ ∧ z_1(σ*) ≠ z_1^a ∧ z_1(σ*) ≠ z_1^b then
    updateBranchingPriorities(σ*), updateConstraintPool()
    Sol ← Sol ∪ {z(σ*)}
    if z_2^a − z_2(σ*) ≥ 2δ_2 then
      I ← I ∪ {[z^a, z(σ*)]}
      Sol ← Sol ∪ BINS(z^a, z(σ*))
    if z_1^b − z_1(σ*) ≥ 2δ_1 then
      I ← I ∪ {[z(σ*), z^b]}
      Sol ← Sol ∪ BINS(z(σ*), z^b)
newSol ← Sol
while newSol ≠ ∅ do
  neighbors ← ∅
  while newSol ≠ ∅ do
    z^a ← pop a solution from newSol
    neighbors ← neighbors ∪ directionalLocBra(z^a)
  Sol ← non-dominated solutions from Sol ∪ neighbors
  newSol ← non-dominated solutions from neighbors
```

We have also tested a variant of this framework in which local branching (using the incumbent solution σ^* as initial solution) is performed whenever an ILP terminates due to the timelimit t_{FP} in the first phase. Thereby, local branching is iteratively applied until a solution is provably optimal for the considered neighborhood size or until a predefined number of maximum iterations is reached. The main idea behind it is to avoid a creation of new intervals based on a rather bad solution σ^*.

It is worth mentioning that we also investigated a heuristic variant of the ϵ-constraint method in which a timelimit is applied to each iteration. If no solution has been identified in the current iteration, a natural idea is to simply decrease ϵ and proceed. Using this strategy may, however, result in an approximate front for which large areas in the objective space remain empty. If on the contrary at least one candidate solution has been found for the current value of ϵ (which may or may not be optimal given a termination due to the timelimit), it is not clear how to set parameter ϵ in the next iteration. Our preliminary experiments with such an ϵ-constraint based heuristics exhibited a rather bad performance (even when combined with local

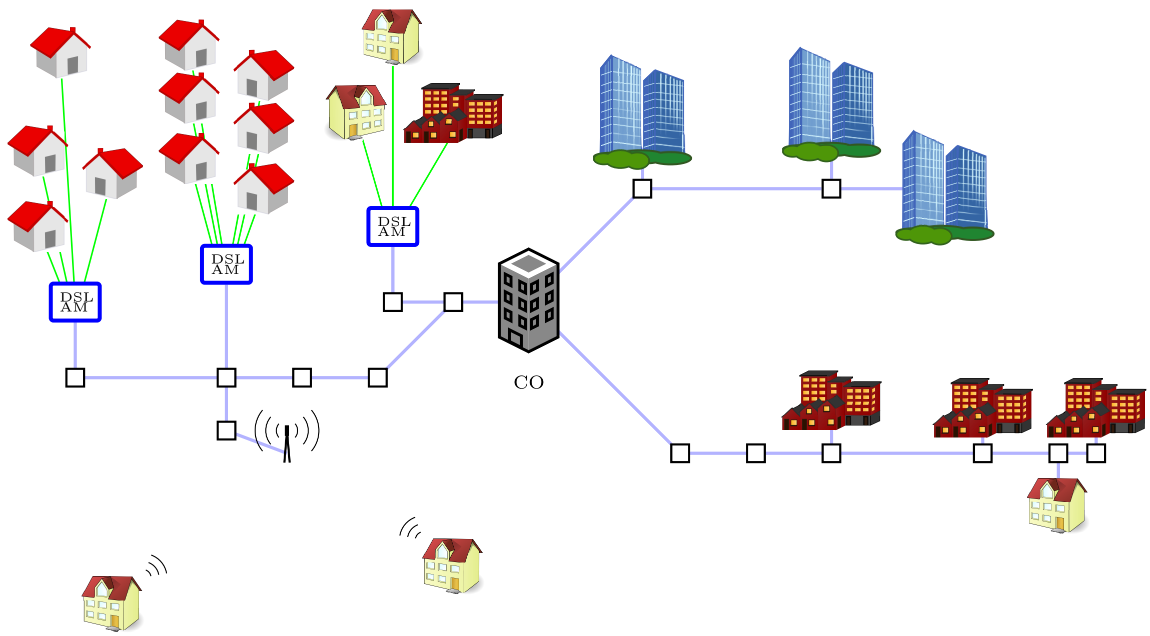


Figure 3: Schematic view of an access network using different technologies (fiber, copper, and wireless); cf. (25)

.

branching) so that we did not try to exploit this idea any further.

4 Bi-objective FTTx Network Design

Our methods are tested on benchmark problems arising in the design of fiber-optic access networks with different potential *technologies* (also called *architectures*), see Figure 3. Despite the fact that a large body of work has been devoted to this topic (see (25) for a recent survey), so-called *mixed deployment strategies* have been considered only recently in (26). In (26), the *two-architecture connected facility location problem* (2AConFL) is introduced, which models the problem of supplying customers of a given deployment area with two potential technologies. *Minimum-coverage rates* are used to specify the fraction of the customers to be supplied by the better and by any of the two technologies, respectively.

A main drawback of modeling the deployment using the 2AConFL is that minimum-coverage rates need to be specified in advance and potentially existing, significantly cheaper solutions slightly violating coverage constraints will never be considered. Since the latter solutions represent attractive options for decision makers, it is desirable to study problem variants in which coverage rates are considered as additional objectives. In order to better capture the trade-off between these two conflicting goals, namely investment costs and achieved coverage rates, we introduce the *multi-objective k-architecture connected facility location problem* (MOkAConFL), generalizing 2AConFL to more than two architectures and to multiple-objectives. After describing the used ILP formulation for MOkAConFL we discuss practically relevant special cases with two objectives including those that will be considered in our computational study.

In an instance of the MOkAConFL we are given a *core graph* $G = (V, E)$ whose node set V is the union of potential *central offices* (COs) Q with opening costs $c_q \geq 0$, $\forall q \in Q$, potential *facility* locations $I = \bigcup_{l=1}^{k} I^l$ per technology l with associated opening costs $c_i^l \geq 0$, $\forall i \in I^l$, $1 \leq l \leq k$, and potential Steiner nodes S. Facilities in this context are multiplexors or switches, and Steiner nodes are, e.g., street-junctions. Edges $e = \{u, v\} \in E$ model potential connections between core nodes u and v and are associated with trenching

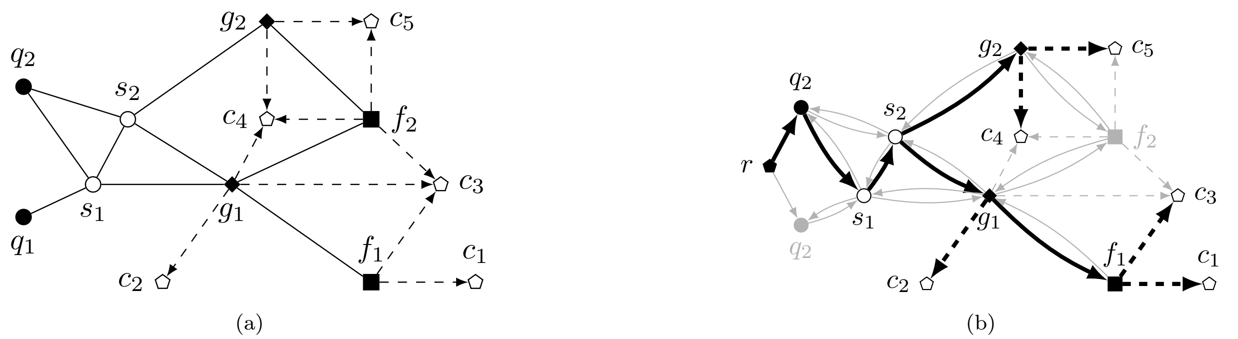


Figure 4: a) An instance of MOkAConFL for $k = 2$ with $Q = \{q_1, q_2\}$, $I^1 = \{f_1, f_2\}$, $I^2 = \{g_1, g_2\}$, and $J = \{c_1, \ldots, c_5\}$. b) An exemplary solution to this instance where each customer is supplied by some architecture and customers c_1 and c_3 are served by the better architecture. The assignment to the artificial root node r determines which *COs* are opened; cf. (26).

costs $c_e \geq 0$. We are further given a set of potential *customers* J with *demands* $d_j \in \mathbb{N}$, $\forall j \in J$, and a bipartite digraph $(I \cup J, \bigcup_{l=1}^{k} A^l)$, $A^l \subseteq I^l \times J$, $1 \leq l \leq k$, modeling potential assignments between facilities and customers using technology l. Thus, each facility in I^l represents a location from which (after the installation of appropriate equipment) some customers can be supplied using architecture l. Note that the sets I^l need not be disjoint.

A facility must be opened if at least one customer is assigned to it and every customer can be assigned to at most one facility. Furthermore, each open facility must be connected to an open central office by a path in the core graph. CO nodes and potential facility locations can be used as Steiner nodes, in which case no opening costs are paid for passing through them. Besides minimizing the overall costs of the network, MOkAConFL aims to maximize the demand served with technology l or better, for $1 \leq l \leq k$ (equivalently, to minimize the demand that is not served with technology l or better). A technology i is considered better as technology j, if its index is smaller, i.e., $i < j$.

The problem is modeled on a digraph $(V \cup J \cup \{r\}, A_r \cup A_c \cup \bigcup_{l=1}^{k} A^l)$. Thereby, r is an artificial root node $r \notin V$ connected to each potential central office $q \in Q$ via arcs $A_r = \{(r, q) \mid q \in Q\}$ that incorporate the corresponding opening costs, i.e., $c_{rq} = c_q$, $\forall q \in Q$. Obviously, if $|Q| = 1$ the creation of the artificial root node and its associated arcs can be skipped and the CO itself can act as the root. The arc set $A_c = \{(u, v) \mid \{u, v\} \in E\}$ is obtained by bi-directing the core edges and we assume that $c_{uv} = c_{vu} = c_e$, $\forall e \in E$. For ease of notation, we use abbreviations $A_{rc} = A_r \cup A_c$, $A_a = \bigcup_{l=1}^{k} A^l$ and $A = A_{rc} \cup A_a$. Figure 4 shows an exemplary instance and a feasible solution.

Our generic ILP model for MOkAConFL, which will be specialized later on, is given by (2)–(11). Thereby, *core arc variables* $x_{ij} \in \{0, 1\}, \forall (i, j) \in A_{rc}$, indicate membership of core and artificial root arcs to the solution while *core node variables* $y_i \in \{0, 1\}, \forall i \in V$, denote if a node i is in the solution. *Assignment arc variables* $x^l_{ij} \in \{0, 1\}, \forall (i, j) \in A^l, 1 \leq l \leq k$, indicate if customer j is supplied by facility i using architecture l, *facility variables* $f^l_i \in \{0, 1\}$, $\forall i \in I^l, 1 \leq l \leq k$, whether or not facility i is open providing connections using architecture l, and *customer variables* $r^l_j \in \{0, 1\}$, $\forall j \in J, 1 \leq l \leq k$, if customer j is supplied by architecture l.

Let $D = \sum_{j \in J} d_j$, and let $I^l_j = \{i \in I^l \mid (i, j) \in A^l\}$ be the set of eligible facilities for a customer $j \in J$ and architecture l, $1 \leq l \leq k$. For a node set $W \subset V \cup J$, let $\delta^-(W) = \{(i, j) \in A \cup A_r \mid i \notin W, j \in W\}$ and $\delta^+(W) = \{(i, j) \in A \cup A_r \mid i \in W, j \notin W\}$ be the set of incoming and outgoing arcs, respectively. Finally, for arc set $\hat{A} \subset A_{rc}$ let $x(\hat{A}) = \sum_{(i,j) \in \hat{A}} x_{ij}$.

$$\min \sum_{(i,j) \in A_{rc}} c_{ij} x_{ij} + \sum_{l=1}^{k} \Big(\sum_{(i,j) \in A^l} c^l_{ij} x^l_{ij} + \sum_{i \in I^l} c^l_i f^l_i \Big) \tag{2}$$

$$
\begin{aligned}
\min \;\; & D - \sum_{m=1}^{l} \sum_{j \in J} d_j r_j^m & 1 \le l \le k & \qquad (3)\\
\text{s.t.} \;\; & \sum_{l=1}^{k} r_j^l \le 1 & \forall j \in J & \qquad (4)\\
& \sum_{i \in I_j^l} x_{ij}^l = r_j^l & \forall j \in J,\ 1 \le l \le k & \qquad (5)\\
& x_{ij}^l \le f_i^l & \forall j \in J,\ i \in I_j^l,\ 1 \le l \le k & \qquad (6)\\
& x(\delta^-(\{i\})) = y_i & \forall i \in V & \qquad (7)\\
& f_i^l \le y_i & \forall\, i \in I_j^l,\ 1 \le l \le k & \qquad (8)\\
& x(\delta^-(W)) \ge y_i & \forall W \subseteq V,\ i \in I \cap W & \qquad (9)\\
& (x^l, f^l, r^l) \in \{0,1\}^{|A^l|+|I^l|+|J^l|} & 1 \le l \le k & \qquad (10)\\
& (x, y) \in \{0,1\}^{|A_{rc}|+|V|} & & \qquad (11)
\end{aligned}
$$

While (2) minimizes the total construction costs of the network, the objectives (3) minimize the demand not served with technology l or better, for $1 \le l \le k$. Constraints (4) and (5) ensure that a unique architecture and assignment arc is used for each connected customer. Inequalities (6) force a facility to be opened whenever an assignment arc issuing from it is chosen. Connectivity constraints (9) (*y-cuts*) ensure, together with constraints (7),(8), that each opened facility is connected to the artificial root node via opened core arcs. Since the root node is adjacent only to the CO nodes it follows that at least one CO node is contained in the solution.

Coverage constraints (12) specifying the minimum fraction p_l, $0 \le p_l \le 1$, of demand to be covered by technology l or better will be added in one of the special cases of MOkAConFL discussed below.

$$\sum_{m=1}^{l} \sum_{j \in J} d_j r_j^m \ge \lceil p_l D \rceil \qquad 1 \le l \le k \qquad (12)$$

If $p_l > 0$ for some l, at least one facility of type l or better must be opened and hence connectivity constraints (9) are strengthened to

$$x(\delta^-(W)) \ge 1, \quad \forall W \subseteq V : \bigcup_{m=1}^{l} I^l \subseteq W \qquad (13)$$

if W contains all potential facilities of type l or better.

Furthermore, whenever all customers must be supplied, inequality (4) is replaced by an equation, which implies that objective (3) is always 0 for $l = k$.

Bi-objective FTTH-Network Design The design of FTTH networks in which customers shall be connected to a central office by fiber-optic cables can be modeled as a bi-objective prize-collecting Steiner tree problem (BOPCSTP), see (22). The goal is to find a solution that minimizes installation costs and at the same time maximizes the percentage of served customers. A transformation to MOkAConFL is obtained by considering only a single architecture ($k = 1$) and no coverage constraints. Facility locations are identical to customer locations in this case.

Bi-objective FTTC-Network Design In FTTC network design, multiplexors that need to be connected to a central office by fiber-optics are installed at certain locations, each potentially serving a set of customers close to it by existing copper cables. Minimizing the network construction costs while minimizing the uncovered customer demand one obtains a bi-objective variant of the connected facility location problem (see, e.g., (17; 18; 26)) which is a special case of MOkAConFL for $k = 1$ and without imposing coverage constraints. We denote this problem as BOConFL.

Bi-objective Two-Architecture Network Design Being concerned with the deployment of two different architectures (e.g., FTTA and FTTC, or FTTC and FTTH) while assuming that one of them is better than the other, the goal is to minimize the resulting network's cost while at the same time minimizing the demand covered with the worse technology. This problem variant, which we denote as BOTAConFL, corresponds to MOkAConFL with $k = 2$ and a given coverage rate p_2 (for technology two or better) suitably chosen in advance by a decision maker.

5 Computational Study

We conduct our computational study on a set of real-world instances representing deployment areas for telecommunication access networks in Germany (with perturbed costs and demands, to ensure data privacy). The main goal of our study is to asses the computational performance of the two new methods, ASOS and the ILP-based heuristic, on a set of realistic instances, and to compare their performance with some of the well-established and well-performing exact methods (according to our recent study in (22)). In addition, we have also implemented a recently proposed rectangular splitting method (6) and included it in our computational study.

Each experiment of our computational study has been performed on a single core of an Intel E5-2670v2 with 2.5 GHz and 64 GB RAM using CPLEX 12.6 for solving (integer) linear programs. A timelimit of 3 600 seconds and a memorylimit of 2 GB each (by setting the `workmem` and `treelim` parameter) has been applied. Furthermore, a timelimit of 60 seconds has been used for BINS, directional local branching and each iteration in the first phase of the heuristic, while the maximum number of local branching iterations in the first phase is set to 5. A total coverage rate of $p_2 = 1$ (i.e., all customers must be connected) has been used for BOTAConFL.

5.1 Branch-and-Cut Configuration

Initialization Since the core-network part of our model (i.e., constraints (7) and (9)) correspond to the well-known cut-formulation of the prize-collecting Steiner tree problem (see, e.g., (27)) we initialize the model by flow-conservation constraints (14) which are known to be strengthening. Additionally, constraints (15) and (16) which cut off the empty solution and forbid cycles of length 2, respectively, are initially included.

$$x(\delta^-(\{i\})) \le x(\delta^+(\{i\})) \qquad \forall i \in V \setminus I \tag{14}$$

$$x(\delta^+(\{r\})) \ge 1 \tag{15}$$

$$x_{ij} + x_{ji} \le y_i \qquad i \in V \tag{16}$$

Separation of Connectivity Constraints For each LP-solution, we first check the constraint pool (see Section 2.3) for violated cuts. In case no cuts are added from the pool, we either search for violated connectivity constraints (9) by computing the connected components of the support graph (if all variable values (x^*, y^*) of the current LP-solution are integral) or by using a maximum-flow algorithm (28) (if some variable values are fractional). In the latter case, backcuts, nested cuts and minimum cardinality cuts as suggested in (16; 27) are used and inequalities are only added if they are violated by at least 0.5. We also avoid to compute the maximum flow to facility nodes that are reachable from r in the subgraph defined by all nodes and arcs whose corresponding variable values are equal to one. Those facilities are identified by a breadth-first search.

Dominated Customer Inequalities We also consider dominated customer inequalities

$$x_{ij}^l \leq \sum_{m=1}^{l} r_{j'}^m, \quad \forall j, j' \in J, i \in I_j^l \cap I_{j'}^l : c_{ij}^l \geq c_{ij'}^l \wedge d_j^l < d_{j'}^l \tag{17}$$

for technology l, which are separated (by enumeration) for integer solutions only. Their validity follows from the fact that customer j (which is dominated by customer j' with respect to facility i and technology l) may only be connected to facility i in an optimal solution, if customer j' is connected using technology l or better.

Branching-Priorities An adaptive branching strategy (cf. Section 2.3) has been used in which priority is given to node and facility opening variables. Their branching priorities are increased by 1 whenever the associated object is contained in a Pareto optimal solution (branching priorities of arc variables are always equal to 0). Initially, we set the branching priorities to 25 (y-variables of nodes that are potential facilities), 20 (facility variables), 15 (y-variables of non-facility nodes), and 5 (customer variables).

Adaptation of the ILP-Heuristics Although our ILP-heuristics BINS and directional local branching described in Section 3 are directly applicable, we slightly adapt them to make use of problem specific knowledge. Notice that once the core nodes, open facilities, and customers are fixed, the considered problems reduce to finding a spanning tree in the core network and an assignment problem in the assignment graph. Thus, only variables associated with the nodes in the core network or with the customers are considered. In the following, we will describe our adaptations for using only node variables of the core network, the adaptations for using only the customer variables work analogously. Let $V(\sigma^i) \subseteq V$ be the core nodes that are selected in solution σ^i. For BINS and given solutions σ^a, σ^b, we fix all variables y_i with $i \in V(\sigma^a) \cap V(\sigma^b)$ to 1 and optionally also fix all variables y_i with $i \in \{V \setminus V(\sigma^a)\} \cap \{V \setminus V(\sigma^b)\}$ to 0. For (directional) local branching and a given solution σ^a, we used asymmetric local branching constraints

$$\sum_{j \in V(\sigma^a)} y_j \geq |V(\sigma^a)| - n, \tag{18}$$

that ensure the selection of at least $|V(\sigma^a)| - n$ core nodes of solution σ^a for a given parameter $n \in \mathbb{N}$.

5.2 Hypervolume Gap Indicator

To better estimate the quality of approximate Pareto frontiers, Boland et al. (6) proposed to compute an upper bound on the area dominated by the optimal frontier which they call *adjusted hypervolume indicator*. Given a set of rectangles $[z^a, z^b]$, where further non-dominated points may lie, their indicator can be computed by adding the area of each such rectangle to the value of the hypervolume indicator.

We now propose a tighter upper bound to which we refer to as *relaxed hypervolume indicator* that also takes into account additional knowledge gained during the course of the current algorithm. Recall that some of our methods allow to conclude that certain areas of a rectangle $[z^a, z^b]$ cannot contain efficient solutions (cf. Section 2.1) or that the rectangle is empty (the latter information is also used in the adjusted hypervolume indicator). Consequently, the area of such a rectangle gives only partial or zero contribution to the relaxed hypervolume indicator.

Definition 1 (Relaxed Hypervolume Indicator, rH)**.** *Let M_i indicate a given solution method M in iteration i and let $\hat{P}_E(M_i)$ be its set of non-dominated points found so far. The relaxed hypervolume indicator, denoted by $rH(M_i)$, is given by the hypervolume $H(\hat{P}_E(M_i))$* plus *the area which may still contain non-dominated points according to the information gained by M up to its current iteration i.*

The relaxed hypervolume indicator is illustrated in Figure 5. Clearly, $H(\hat{P}_E(M_i)) \leq H(P_E) = rH(M_T) \leq rH(M_i)$ holds for any iteration i smaller than or equal to the final iteration T in which M terminates (assuming that M discovers the complete Pareto front). Thus, we introduce the *hypervolume gap indicator* as follows.

Definition 2 (Hypervolume Gap Indicator, gH)**.** *Let $\hat{P}_E(M_i)$ be the set of non-dominated points found up to iteration i, using method M. The hypervolume gap indicator for M in iteration i is*

$$gH(M_i) = \frac{rH(M_i)}{H(\hat{P}_E(M_i))} - 1.$$

By definition we have $gH(M_i) \geq 0$ and $gH(M_i) = 0$ iff $i = T$, i.e., when M has identified the provably optimal Pareto front. Note that the adjusted hypervolume indicator (6) coincides with rH for the rectangle-splitting method proposed in (6). On the other hand, as it will be described below, rH provides tighter bounds for ASOS and BSOS.

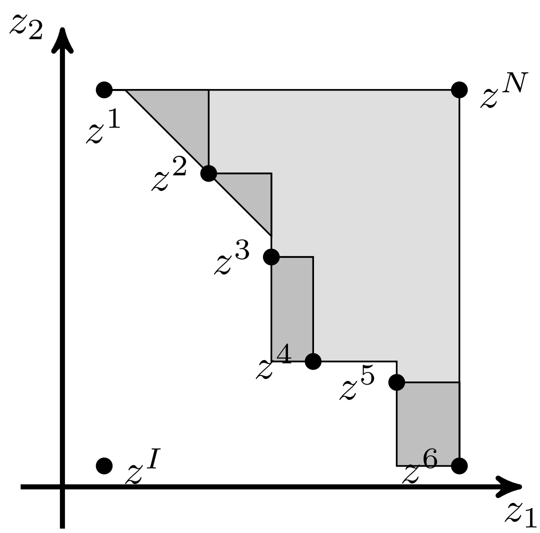
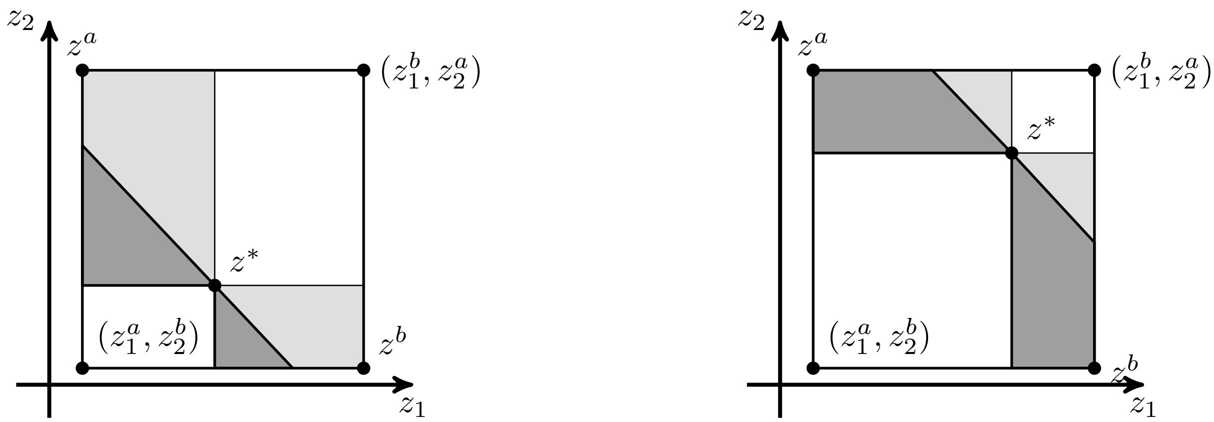


Figure 5: A set of discovered Pareto optimal solutions $\hat{P}_E = \{z^1, \ldots, z^6\}$ in the current iteration i. The lightly shaded area defines the hypervolume $H(\hat{P}_E)$ and the union of both shaded areas defines the relaxed hypervolume $rH(M_i)$. It is assumed that M_i implies that the rectangle $[z^4, z^5]$ does not contain non-dominated solutions and that only a partial area of the rectangles $[z^1, z^2]$ and $[z^2, z^3]$ may contain further non-dominated solutions. Observe that, e.g., BSOS or ASOS can derive both conclusions.

Shrinking of Rectangles and Computation of rH for ASOS/BSOS Consider a rectangle $[z^a, z^b]$ and a new solution z^* found inside the rectangle. As observed in (22, Prop. 1), the two new rectangles $[z^a, z^*]$ and $[z^*, z^b]$, can be shrinked, depending on the weights ω_1, ω_2 used to determine z^*. Let $z^\omega = \omega_1 z_1^* + \omega_2 z_2^*$. The shrinking depends on the position of the line $\{z \in [z^a, z^b] \mid \omega_1 z_1 + \omega_2 z_2 = z^\omega\}$ within the rectangle $[z^a, z^b]$; clearly, below this line in $[z^a, z^b]$, there are no non-dominated points. Therefore, only areas above that line in the rectangles $[z^a, z^*]$ and $[z^*, z^b]$ are contributing to rH. This is the main difference to the definition of the adjusted hypervolume indicator (6), that would consider the whole areas of $[z^a, z^*]$ and $[z^*, z^b]$. If ω_1, ω_2 are chosen in such a way that the level lines of the objective are parallel to the line through z^a and z^b, we distinguish between the following two cases: Point z^* corresponds either to a supported solution (case A, Figure 6), or to a non-supported one (case B, Figure 6). Notice that in case A, the area that contributes to rH is the area of two trapezoids, and in case B, it is the area of two triangles. If ω_1, ω_2 are chosen differently (as e.g., in Lemma 1), we may end up with calculating a sum of a trapezoid and a triangle.

Observe that after recursively applying this procedure for subsequent rectangles, one may end up with arbitrary convex polygons whose area is not easy to calculate. We therefore overestimate these areas by only considering trapezoids and triangles. With this calculation, it may happen that $rH(M_i) > rH(M_{i+1})$, but $H(P_E) = rH(M_T)$ holds.



(a) Case A: z^* is a supported non-dominated point.

(b) Case B: z^* is a non-supported non-dominated point.

Figure 6: Two possible cases of calculation of the value contributing to rH by a rectangle. Weights ω_1, ω_2 are assumed to be chosen in such a way that the level lines of the objective are parallel to the line between z^a and z^b. The area contributing to rH is shaded in light gray. The dark shaded area is not contributing to rH.

5.3 Instances

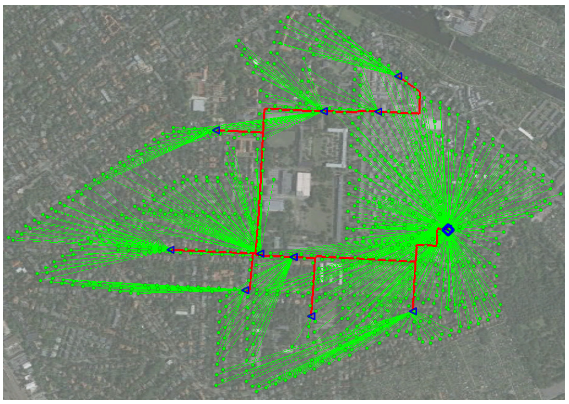
Our benchmark instances are based on realistic networks representing FTTH/FTTC deployment areas in Germany. Five deployment areas are considered: `berlin-tu`, `berlin-rotdorn`, `vehlefanz`, `atlantis` and `berlin-lichterfelde`, generating five groups of instances. Within each group, we consider different scenarios regarding the possible distribution of existing copper infrastructure, resulting in instances with similar network topology within the same group, but with different distribution of potential facilities, and different assignment graphs. These instances have been partly used in (26). Basic properties of our instances are summarized in Table 1. Facilities of technology one represent FTTH connections, which means that each customer location is at the same time a potential facility location (for FTTH technology). Hence, customer nodes have degree one for the FTTH technology (i.e., $|I^1| = |C| = |A^1|$ holds). Facilities of technology two represent FTTC connections, i.e., there are at least two customers, which can be assigned to such a facility.

Table 1: Details of the test instances. # gives the number of instances within each group, *abbrev.* gives the abbreviation used for the name of the group, $|I^1|$ and $|I^2|$ are the number of facilities of technology one and two, resp., $|S|$ is the number of Steiner nodes, $|C|$ is the number of customers, $|CO|$ is the number of available central offices, $|E|$ is the number edges in the core graph, and $|A^1|$, $|A^2|$ are the numbers of assignment arcs.

set	*abbrev.*	#	$\|I^1\|, \|C\|, \|A^1\|$	$\|I^2\|$	$\|S\|$	$\|CO\|$	$\|E\|$	$\|A^2\|$
`berlin-tu`	T	19	39	15-70	271-326	4	560	45-211
`berlin-rotdorn`	R	14	91	15-66	95-146	2	314	107-502
`vehlefanz`	V	54	238	28-169	483-624	5	1096	306-3531
`atlantis`	A	16	345	16-102	550-636	4	1029	506-2607
`berlin-lichterfelde`	L	12	747	79-446	618-985	5	2074	1117-7260

For BOConFL, only FTTC facilities are available, while for BOTAConFL, both FTTC and FTTH facilities can be installed, the latter ones representing the better technology. Note that for BOTAConFL, for three instances from `berlin-lichterfelde`, the boundary points of the Pareto front could not be determined within the given runtime. Thus, we consider 115 instances in total for BOConFL and 112 for BOTAConFL.

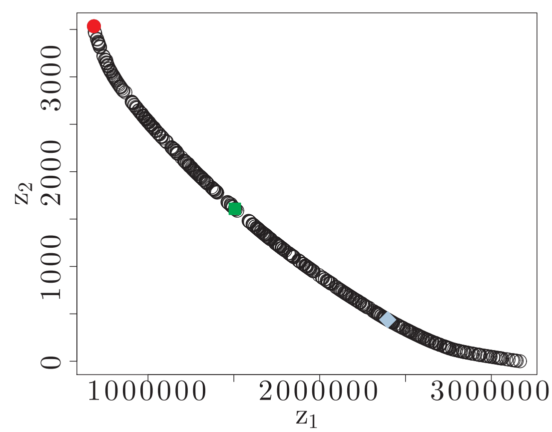
Figures 7a to 7c illustrate three Pareto optimal solutions for one of the largest instances from our benchmark set when considering the mixed deployment (BOTAConFL). In the first solution, most of the customers are connected using the FTTC technology, while in the third solution the situation is the opposite, most of the customers are connected using the FTTH technology. It can be observed that some of FTTC facilities are open in all three solutions, while others are only open in one or two of the solutions. Such an analysis, determining which features of a deployment are occurring in many Pareto optimal solutions, can be a helpful guidance for decision makers. Figure 7d shows the Pareto front obtained by our ILP-based heuristic with the three pictured solutions highlighted.



(a) Pareto optimal solution, $z_1 = 684801$, $z_2 = 3534$

(b) Pareto optimal solution, $z_1 = 1505710$, $z_2 = 1608$

(c) Pareto optimal solution, $z_1 = 2396450$, $z_2 = 438$



(d) Approximate Pareto front discovered by our ILP-based heuristic method

Figure 7: Instance from the set `berlin-lichterfelde`. (a)-(c): three Pareto optimal solutions, (d): approximate Pareto front discovered by our ILP-based heuristic (which worked best for this instance). Customers connected with the FTTH technology are given as orange circles, customers connected with FTTC are given as green circles. Opened facilities are given as blue triangles, the opened central office is the blue diamond. Solution (a) is indicated as red circle in the Pareto front, solution (b) as green rectangle and solution (c) as blue diamond.

5.4 Results

The purpose of our computational study was to assess the efficacy of the new exact method (abbreviated as *asos* below) and the new two-phase ILP-based heuristic (denoted by *ilph*). To this end, we have also implemented the ϵ-constraint method (*eps*) and BSOS (*bsos*). According to our recent study on BOPCSTP, *eps* and *bsos* computationally outperformed other iterative exact methods (22). We additionally consider the rectangle-splitting method (*rect*) recently proposed in (6). If a method is combined with BINS (which turned out to be beneficial in most of the cases, see below), a letter B is added to the name of the method, e.g., *bsosB* means binary search with BINS. Let $\mathcal{M}$ denote the set of all methods considered in this study, i.e., $\mathcal{M} = \{asos, asosB, bsos, bsosB, eps, ilph, rect, rectB\}$.

Default Implementation Settings Preliminary tests led us to use the configuration described in the following for our main computations. Dominated customer inequalities (17) reduced the number of weakly-dominated points discovered; however, the resulting increase in runtime for a single iteration led us to turn them off. All local branching neighborhoods worked very well; we decided to use the neighborhood defined by customer variables whose values are equal to one and radius $n = 10$. No clear picture emerged regarding the attempt to prove optimality of more than one solution in an iteration (application of Lemma 1). For some instances it payed off, while for others the changed objective function coefficients increased the difficulty of solving the ILPs. Thus, we finally decided not to use this setting. Directional local branching is only used in the second phase of *ilph* (as described in Section 3.3). The variant of *ilph* with local branching also applied in the first phase performed very similar to *ilph*.

Comparison Based on Hypervolumes We start our comparison by showing performance plots depicting the number of instances against the square root of the hypervolume gap: Figures 8 and 9 report the obtained hypervolume gaps for BOConFL and BOTAConFL, respectively. Four methods are compared: *eps*, *rectB*, *asosB* and *bsosB*. In the remainder of this section, the hypervolume gap for a method $M \in \mathcal{M}$ is calculated as:

$$gH(M) = \frac{rH}{H(\hat{P}_E(M))} - 1, \quad \text{where} \quad rH = \min_{M \in \mathcal{M}} rH(M). \tag{19}$$

In other words, for the hypervolume gap calculation, the best rH over all methods is used. Most of the time, the best rH is achieved by *bsosB* or *asosB*, since both *eps* and *rectB* often exceed the time- or memorylimit in an iteration where most of the front still remains undiscovered. The square root-transformation is chosen to improve the readability of the plots.

We notice that the worst hypervolume gaps are by far provided by *eps*, followed by *rectB*, while the remaining two methods (*bsosB* and *asosB*) provide hypervolumes of similar quality. The latter effect can possibly be explained by the weak performance of *eps*. Consequently, *asosB* (which combines *eps* and *bsosB*) cannot draw significant advantages from *eps*. We also observe that the considered methods establish the complete Pareto front for 40 to 55 instances (out of 115) for BOConFL and for 20 to 42 instances (out of 112) for BOTAConFL.

Figures 10 and 11 compare *bsos* and *asos* with and without BINS, for BOConFL and BOTAConFL, respectively, without the 60 individually easiest instances for each method. We observe that the use of BINS clearly improves the performance (with respect to the obtained hypervolume gaps) both for BOConFL and for BOTAConFL on these hard instances that cannot be solved withing the given time- or memorylimit.

Drawbacks of Analysis Based on Hypervolumes A significant drawback of analyzing bi-objective methods by means of the hypervolume is that this indicator does not tell a lot about the number of non-dominated points discovered by a method and their distribution in the objective space. A method may, for example, provide a very small hypervolume gap even if it discovers only very few non-dominated points. This is demonstrated in Figure 12 where we plot the set of non-dominated points obtained by four methods: *eps*, *rectB*, *asosB* and *ilph* for instance A3 for BOConFL. One observes that the most useful approximation of the Pareto front is obtained by *ilph*. It is, however, almost impossible to distinguish between *rectB*, *asosB*,

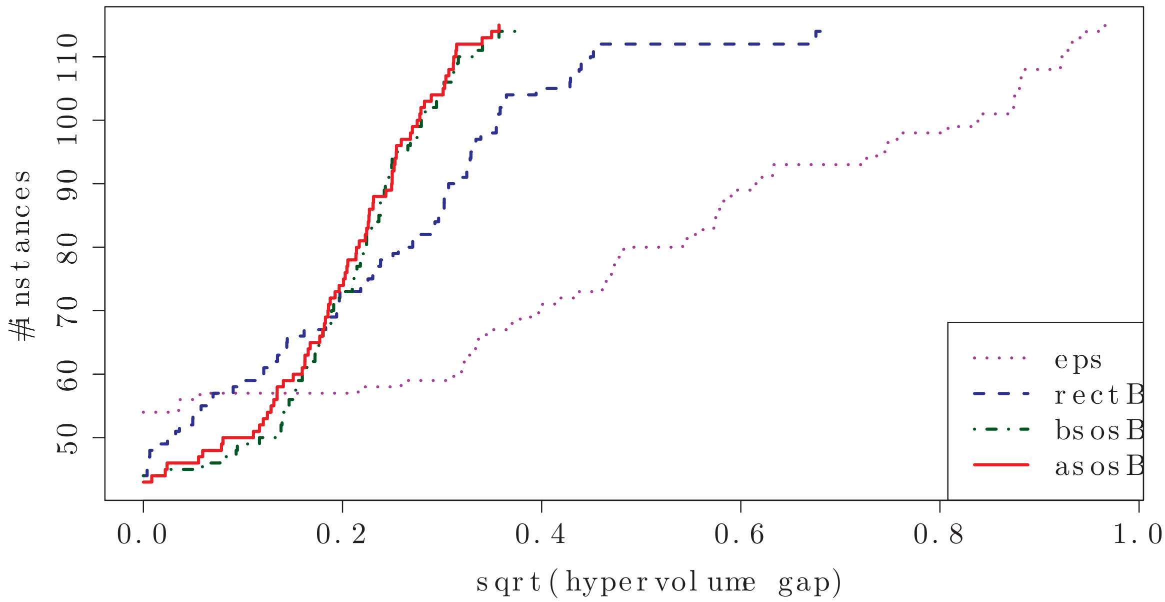


Figure 8: Exact methods applied to BOConFL.

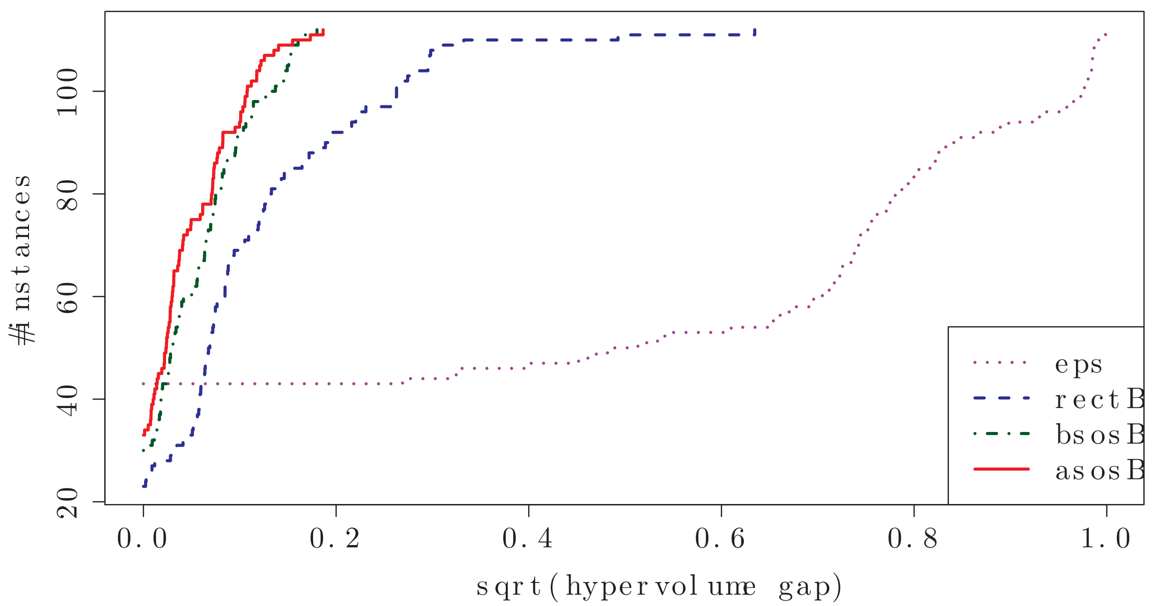


Figure 9: Exact methods applied to BOTAConFL.

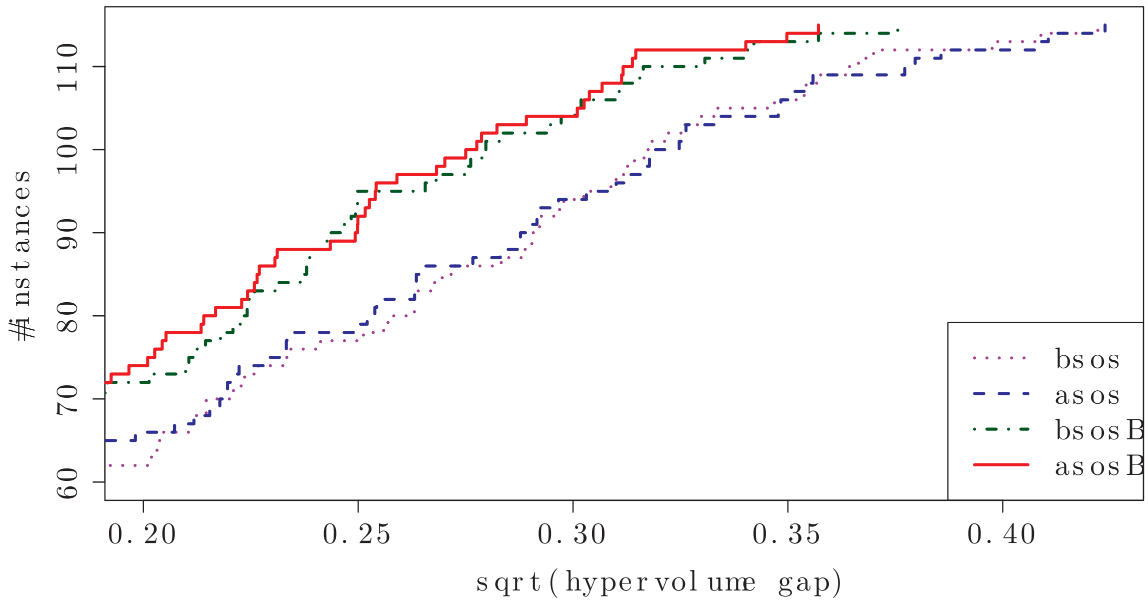


Figure 10: *asos* and *bsos* with and without BINS applied to BOConFL.

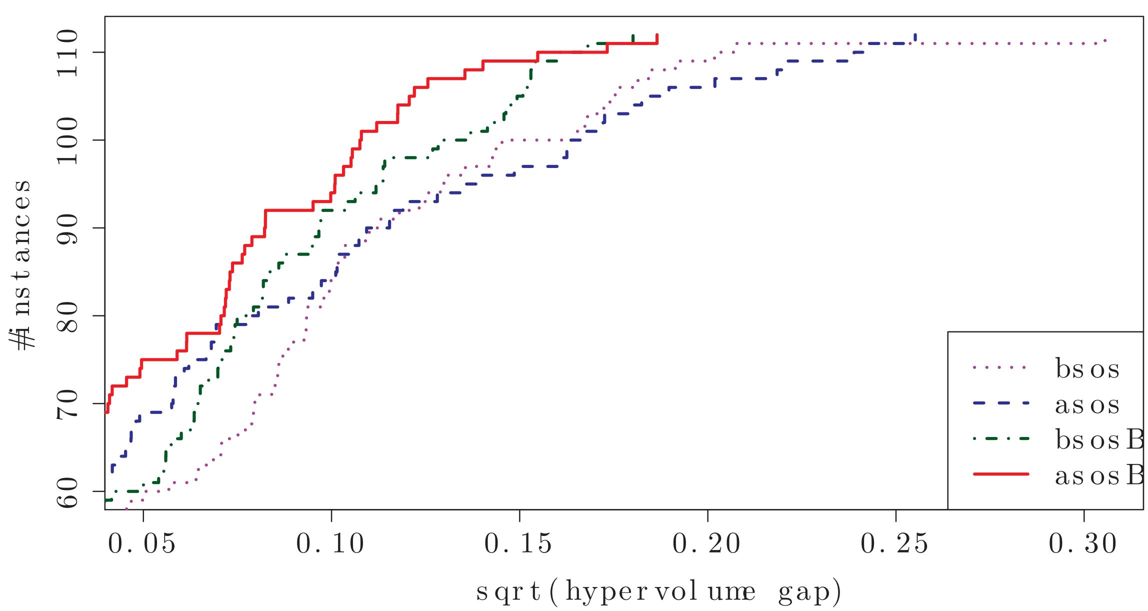
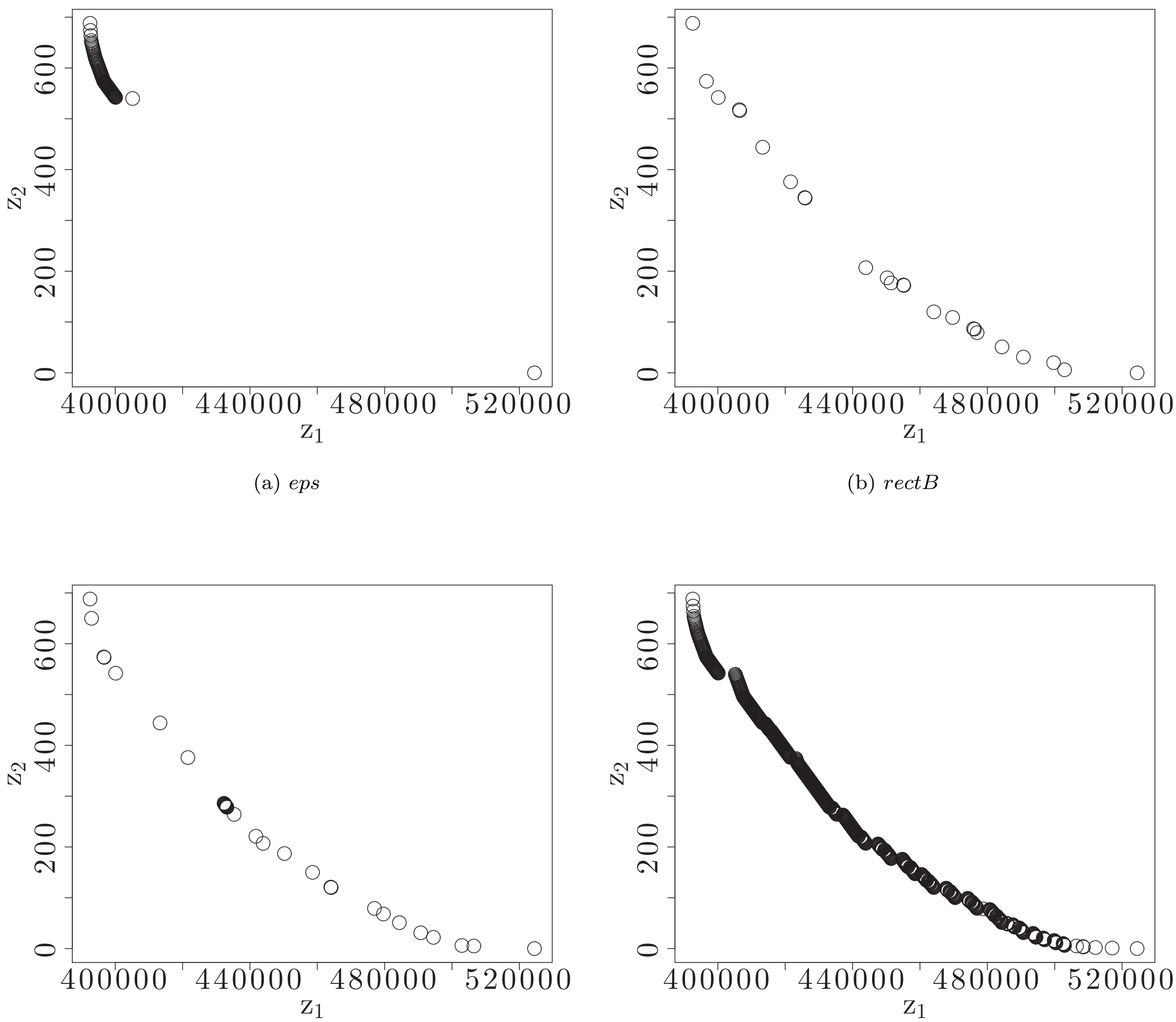


Figure 11: *asos* and *bsos* with and without BINS applied to BOTAConFL

and *ilph* when comparing the values of the hypervolume; see Figure 13 which shows the hypervolume versus runtime for the same instance. In the remainder of our computational study, we will therefore provide a more detailed comparison of selected methods, which (besides the hypervolume gaps) also details numbers of non-dominated points discovered and/or proved to be Pareto optimal.



(a) *eps*

(b) *rectB*

(c) *asosB*

(d) *ilph*

Figure 12: Pareto fronts discovered by different methods within the given memory-and timelimit.

A More Detailed Computational Analysis Figure 13 depicts a typical progress of the hypervolume which occurs for many of our more difficult benchmark instances. We observe that, for the given instance, only one exact method (*rectB*) terminates due to the timelimit while the other two (*eps* and *asosB*) exceed the memorylimit. Method *ilph* terminates since no new solution could be found. Contrary to the exact methods, the heuristic is successfully prevented from getting stuck while calculating a single Pareto optimal solution, due to the timelimit imposed in each iteration.

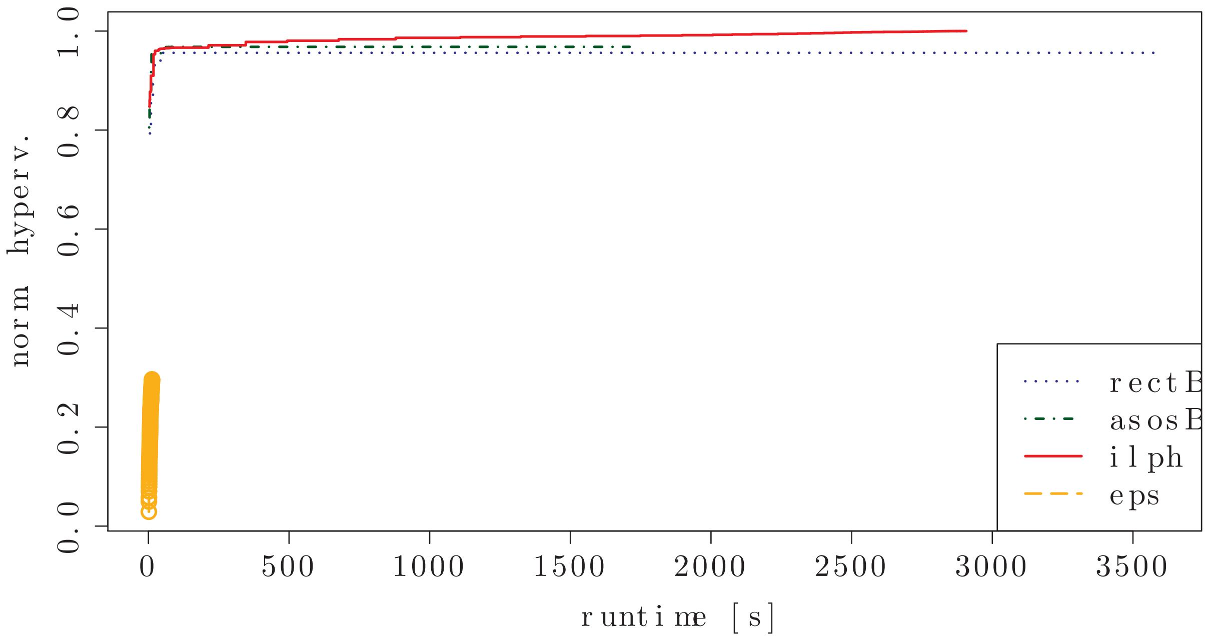


Figure 13: Normalized hypervolume against runtime for different methods and instance A3 for BOConFL.

We also see that the ϵ-constraint method hits the memorylimit quite early and therefore derives only a very limited part of the Pareto front and a small hypervolume. This behavior may be not too critical in case we are lucky and a decision maker is only interested in solutions found in the considered region. This outcome is, however, clearly not desirable in case it is not known beforehand which area of the objective space is relevant.

For this reason, the goal in designing our heuristic framework was to provide many (possibly Pareto optimal) solutions that also cover most of the Pareto front. We next analyze whether this goal has been achieved and also discuss the effects of BINS and the two phases. Thereby, $ilph - nobins$ is used to denote the variant of the heuristic framework without BINS.

Table 2 details the performance of *ilph* and $ilph - nobins$ for BOConFL on instances for which the complete Pareto front could be identified using the ϵ-constraint method (i.e., all instances from set T except for T5). In addition, results for selected difficult instances of sets A and R for BOConFL (A1, R5, R9) and set A for BOTAConFL (A11, A13, A14, A16)[1] are given. Besides the number of Pareto optimal solutions of $ilph-nobins$ and *ilph* in their first and second phase ($|\mathcal{Z}_1^*|$, $|\mathcal{Z}_2^*|$), we also report the corresponding runtimes (t_1, t_2), the sizes of the obtained heuristic fronts ($|\mathcal{Z}_1|$, $|\mathcal{Z}_2|$), and their associated hypervolume gaps (gH_1, gH_2).

We note that the group T for BOConFL has been the easiest in our benchmark set, and the ϵ-constraint method managed to solve all these instances (except T5). From Table 2 we observe that both with and without BINS, *ilph* works very well and BINS gives a significant improvement in the number of non-dominated points discovered in the first phase, while the additional cost on runtime is quite modest. Naturally, for this easy group T, *eps* outperforms the two variants of the heuristic framework with respect to the required

[1] To obtain the results for *eps* for the latter instances, we set the timelimit to 40 hours and the memorylimit to 60GB. Note that even with these limits, some instances from these sets could not be solved with *eps* (and the sets V and L are even more difficult to solve)

runtime. We observe that BINS helps the ILP-heuristic to find between 25% to 50% of the non-dominated points in its first phase within 1/3 of the total running time of *eps*. Except for one instance, the hypervolume gap of *ilph* is below 0.05% after the first phase and the whole Pareto front is discovered after the second phase for most instances (for T16 and T19 the front is determined already in the first phase). We also observe that both after the first and second phase, the heuristic fronts contain mostly points that are in fact non-dominated.

Turning our attention to the more challenging instances A1, R5, R9 (BOConFL) and A11, A13, A14, A16 (BOTAConFL), it can be seen that the fraction of discovered non-dominated points, and also the hypervolume gap, are very similar to the respective values in the easier instances. The runtime of the first phase is much smaller than the runtime of *eps*: At the end of the second phase more than 66% of the Pareto front is discovered, while the time spent is less than half the time *eps* needed to discover the whole front. We want to point out the results obtained for R9 (BOConFL) where both variants of the heuristic manage to discover the whole front in less than 220 seconds, while *eps* requires more than 17000 seconds for the same task.

Table 2: Overview of runtime and size of Pareto front for *eps*, the ILP-based heuristic (*ilph*) and its variant without BINS ($ilph - nobins$). Instances T1-T19, A1, R5, R9 are for BOConFL; and A11, A13, A14, A16 are for BOTAConFL. $|\mathcal{Z}|$ is the size of the Pareto front, $t[s]$ is the runtime for *eps*, $|\mathcal{Z}_1^*|$ shows, how many solutions on the heuristic front of the first phase are really Pareto optimal, $|\mathcal{Z}_1|$ denotes the size of the heuristic front after the first phase, $|\mathcal{Z}_2^*|$ and $|\mathcal{Z}_2|$ are the resp. indicators for the second phase. The hypervolume gap after the first, resp. second phase is denoted by $gH_1[\%]$, resp. $gH_2[\%]$ The runtime for the first phase is indicated with $t_1[s]$ and the total runtime (i.e., first and second phase) with $t_2[s]$.

	eps		$ilph - nobins$								*ilph*							
inst	$\|\mathcal{Z}\|$	$t[s]$	$\|\mathcal{Z}_1^*\|$	$\|\mathcal{Z}_1\|$	$\|\mathcal{Z}_2^*\|$	$\|\mathcal{Z}_2\|$	$gH_1[\%]$	$gH_2[\%]$	$t_1[s]$	$t_2[s]$	$\|\mathcal{Z}_1^*\|$	$\|\mathcal{Z}_1\|$	$\|\mathcal{Z}_2^*\|$	$\|\mathcal{Z}_2\|$	$gH_1[\%]$	$gH_2[\%]$	$t_1[s]$	$t_2[s]$
T1	206	584	16	18	199	204	0.0478	0.0001	27	1185	36	42	193	202	0.0219	0.0004	38	1107
T2	178	498	14	15	166	174	0.0657	0.0005	16	929	30	33	173	176	0.0376	0.0003	22	914
T3	120	2357	18	21	114	119	0.0360	0.0003	78	1749	32	32	111	117	0.0152	0.0005	84	1622
T4	133	370	21	22	121	130	0.0357	0.0006	24	689	37	40	126	128	0.0175	0.0001	36	629
T6	139	247	19	20	136	139	0.0285	0.0000	13	402	33	35	135	138	0.0210	0.0001	19	385
T7	122	146	16	16	122	122	0.0753	0.0000	7	234	28	29	122	122	0.0364	0.0000	10	263
T8	81	93	21	21	81	81	0.0783	0.0000	4	103	29	30	81	81	0.0667	0.0000	6	127
T9	72	172	26	28	72	72	0.0162	0.0000	30	198	36	36	72	72	0.0139	0.0000	38	193
T10	76	91	28	28	76	76	0.0080	0.0000	13	127	39	41	76	76	0.0034	0.0000	16	130
T11	62	40	27	27	62	62	0.0215	0.0000	5	70	37	37	62	62	0.0045	0.0000	7	71
T12	76	57	25	25	76	76	0.0316	0.0000	4	87	36	36	76	76	0.0044	0.0000	6	103
T13	71	43	25	25	71	71	0.0341	0.0000	4	64	37	38	71	71	0.0274	0.0000	6	94
T14	69	145	24	24	69	69	0.0215	0.0000	14	209	36	36	69	69	0.0232	0.0000	12	260
T15	59	33	27	27	59	59	0.0247	0.0000	4	48	38	39	59	59	0.0028	0.0000	7	49
T16	63	32	27	28	63	63	0.0076	0.0000	8	75	40	41	63	63	0.0021	0.0000	12	83
T17	40	3	25	25	40	40	0.0034	0.0000	3	7	40	40	40	40	0.0000	0.0000	4	9
T18	217	470	17	19	211	216	0.0719	0.0001	22	1405	31	44	212	218	0.0399	0.0001	30	1401
T19	39	3	23	23	39	39	0.0288	0.0000	2	6	39	39	39	39	0.0000	0.0000	4	8
A1	672	5064	57	67	308	387	0.0269	0.0069	213	3600	89	123	299	391	0.0171	0.0067	341	3600
R5	271	14834	27	30	249	251	0.0225	0.0003	116	3600	61	64	249	253	0.0143	0.0003	163	3600
R9	121	17470	15	15	121	121	0.0221	0.0000	2	166	86	86	121	121	0.0048	0.0000	7	217
A11	597	24697	93	101	365	481	0.0166	0.0044	101	3600	247	283	453	544	0.0074	0.0013	242	3600
A13	611	9162	86	95	450	538	0.0222	0.0025	70	3600	204	236	442	541	0.0149	0.0019	179	3600
A14	608	16878	86	94	408	528	0.0259	0.0031	68	3600	209	241	488	549	0.0137	0.0018	166	3600
A16	620	21533	85	96	358	477	0.0231	0.0050	104	3600	203	240	439	532	0.0173	0.0023	245	3600

Next, we provide detailed computational results to compare our methods *asosB* and *ilph* against *eps* and *rectB*. Tables 3 to 6 report results obtained for BOConFL, whereas Tables 7 to 10 the results obtained for BOTAConFL. Recall that each method returns at the end a set of non-dominated points, some of them being provably non-dominated, others being heuristically obtained. For every instance we present value $|\mathcal{Z}^{opt}|$, which is either the size of the Pareto front, if at least one method managed to discover it, or the size of the union of the points that were proven to be non-dominated by at least one of the methods, otherwise. If at least one method managed to discover the whole Pareto front, this is indicated by a bold value in the column $|\mathcal{Z}^{opt}|$. Moreover, column $|\mathcal{Z}^{*}|$ denotes the number of points proven to be non-dominated by the respective method. Thus, $|\mathcal{Z}^{opt}| = |\bigcup_{M\in\mathcal{M}} Z^*_M|$, where Z^*_M denotes the set of provably non-dominated points discovered by method $M \in \mathcal{M}$. In addition, $|\mathcal{Z}^{+}|$ indicates the number of discovered non-dominated points from $\mathcal{Z}^{opt}$ for which the corresponding method did not prove that they are non-dominated (i.e., heuristically identified points). The number of remaining non-dominated points, contained in the final set produced by a method is shown in the column $|\mathcal{Z}^{-}|$.[2] In addition, we also report the runtime ($t[s]$) and hypervolume gap ($gH[\%]$) calculated according to (19)for each method. By TL or ML in column $t[s]$ we denote if a method reached the time- or memorylimit, respectively. A zero hypervolume gap is indicated by opt in the column $gH[\%]$. Observe that $gH[\%]$ can be zero even when the method terminates due to the time- or memorylimit. This happens when only rectangles containing no further non-dominated points remain open, but proving emptiness was not possible during the given limits. The best value for a given instance in this column is marked in bold.

From Tables 3 to 6 (i.e., for BOConFL) we conclude that the ϵ-constraint method only works well for the smallest instance groups T and R (cf. Table 3). As the instances become larger, *eps* is outperformed by the other methods. For example, for half of the instances of the set V (cf. Tables 4 to 5), the hypervolume gap obtained by *eps* is above 0.2%. At the same time, the gaps obtained by *asosB* and *rectB* are (except for a few cases) consistently below 0.1%. The best-performing method for the set V, with respect to both the number of discovered non-dominated points (i.e., $|\mathcal{Z}^{*}|$ plus $|\mathcal{Z}^{+}|$) as well as the hypervolume gap, is *ilph*. The hypervolume gap is by one order of magnitude smaller than the one by *eps*, and the largest hypervolume gap of *ilph* for the set V is 0.077%. A similar picture emerges for the remaining two sets of larger benchmark instances, namely A and L.

For BOTAConFL (see Tables 7 to 10) we again observe that for the easier instance groups T and R, all methods work well and almost always find the complete Pareto front, with *eps* being the fastest. For the set V, *asosB* is the overall best method, in particular with respect to the number of discovered non-dominated points (with an exception of some outliers for which *eps* manages to determine the whole front). For the two largest sets A and L, again, our ILP heuristic *ilph*, provides the best results, and for the set L containing the largest instances, *ilph* and *asosB* work best, both in terms of number of points and hypervolume gap.

Overall, we conclude the following: for easier instances, where the underlying branch-and-cut algorithm runs fast and stable, *eps* is the best performing method. As the size/difficulty of instances increases, *eps* tends to get stuck due to excessive memory or time usage. Alternative iterative methods, like *asos*, *bsos* or *rect*, studied in this paper, can better deal with these issues, providing significantly smaller hypervolume gaps. However, we demonstrate that looking solely at the hypervolume (gaps) is not sufficient, as the number of Pareto optimal points discovered by a method can still be relatively small. With this respect, our study shows that the method that provides the most accurate, diverse and rich Pareto fronts is the new ILP-heuristic *ilph*.

[2]These points can be of two different types: i) they are dominated by some of the points on the Pareto frontier $\mathcal{Z}^{opt}$, ii) they are not dominated by any of the points on the Pareto front $\mathcal{Z}^{opt}$, i.e., their corresponding solutions could be Pareto optimal but no method managed to prove it. The latter case, obviously, can only happen for instances, where no method discovered the whole Pareto front.

Table 3: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\lvert\mathcal{Z}^{opt}\rvert$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$
T1	**206**	206	0	0	584	**opt**	206	0	0	1868	**opt**	206	0	0	1022	**opt**	13	180	9	1107	0.000
T2	**178**	178	0	0	498	**opt**	178	0	0	2073	**opt**	178	0	0	687	**opt**	16	157	3	914	0.000
T3	**120**	120	0	0	2357	**opt**	115	2	0	TL	0.000	120	0	0	1017	**opt**	19	92	6	1622	0.000
T4	**133**	133	0	0	370	**opt**	133	0	0	1144	**opt**	133	0	0	538	**opt**	23	103	2	629	0.000
T5	105	61	1	3	ML	0.159	13	10	4	TL	0.038	62	11	11	TL	**0.006**	19	47	35	3444	0.007
T6	**139**	139	0	0	247	**opt**	139	0	0	835	**opt**	139	0	0	399	**opt**	21	114	3	385	0.000
T7	**122**	122	0	0	146	**opt**	122	0	0	471	**opt**	122	0	0	238	**opt**	19	103	0	263	**opt**
T8	**81**	81	0	0	93	**opt**	81	0	0	199	**opt**	81	0	0	119	**opt**	25	56	0	127	**opt**
T9	**72**	72	0	0	172	**opt**	72	0	0	458	**opt**	72	0	0	166	**opt**	32	40	0	193	**opt**
T10	**76**	76	0	0	91	**opt**	76	0	0	255	**opt**	76	0	0	148	**opt**	33	43	0	130	**opt**
T11	**62**	62	0	0	40	**opt**	62	0	0	89	**opt**	62	0	0	56	**opt**	32	30	0	71	**opt**
T12	**76**	76	0	0	57	**opt**	76	0	0	156	**opt**	76	0	0	86	**opt**	31	45	0	103	**opt**
T13	**71**	71	0	0	43	**opt**	71	0	0	123	**opt**	71	0	0	68	**opt**	31	40	0	94	**opt**
T14	**69**	69	0	0	145	**opt**	69	0	0	500	**opt**	69	0	0	160	**opt**	31	38	0	260	**opt**
T15	**59**	59	0	0	33	**opt**	59	0	0	69	**opt**	59	0	0	50	**opt**	34	25	0	49	**opt**
T16	**63**	63	0	0	32	**opt**	63	0	0	118	**opt**	63	0	0	76	**opt**	36	27	0	83	**opt**
T17	**40**	40	0	0	3	**opt**	40	0	0	8	**opt**	40	0	0	6	**opt**	40	0	0	9	**opt**
T18	**217**	217	0	0	470	**opt**	217	0	0	1845	**opt**	217	0	0	849	**opt**	15	197	6	1401	0.000
T19	**39**	39	0	0	3	**opt**	39	0	0	8	**opt**	39	0	0	5	**opt**	39	0	0	8	**opt**
R1	**413**	413	0	0	550	**opt**	346	45	0	ML	0.000	413	0	0	447	**opt**	28	385	0	444	**opt**
R2	**300**	300	0	0	715	**opt**	299	1	0	TL	**opt**	300	0	0	2452	**opt**	40	258	2	712	0.000
R3	**264**	264	0	0	1730	**opt**	109	57	4	ML	0.001	35	11	6	ML	0.018	43	219	1	975	0.000
R4	**292**	292	0	0	944	**opt**	197	51	0	TL	0.000	263	9	0	TL	0.000	40	252	0	579	**opt**
R5	262	120	0	0	ML	0.187	51	38	12	TL	0.004	127	13	15	ML	0.003	44	196	13	TL	**0.001**
R6	113	66	0	1	ML	0.212	17	9	1	ML	0.038	42	3	4	ML	0.039	65	47	53	720	**0.024**
R7	93	72	0	0	ML	0.143	5	2	0	ML	0.184	22	5	2	ML	0.077	72	19	39	553	**0.047**
R8	**123**	123	0	0	176	**opt**	14	8	1	ML	0.021	97	10	0	TL	0.000	82	41	0	218	**opt**
R9	107	82	0	0	ML	0.067	27	11	2	ML	0.016	61	6	2	ML	0.014	83	24	14	217	**0.008**
R10	**133**	133	0	0	22	**opt**	133	0	0	247	**opt**	133	0	0	1194	**opt**	82	51	0	91	**opt**
R11	**120**	120	0	0	98	**opt**	13	9	0	ML	0.018	40	8	0	TL	0.016	82	38	0	178	**opt**
R12	**124**	124	0	0	106	**opt**	124	0	0	206	**opt**	124	0	0	421	**opt**	84	40	0	153	**opt**
R13	**457**	457	0	0	382	**opt**	457	0	0	852	**opt**	457	0	0	422	**opt**	32	424	1	596	0.000
R14	**91**	91	0	0	4	**opt**	91	0	0	17	**opt**	91	0	0	11	**opt**	91	0	0	16	**opt**

Table 4: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\|\mathcal{Z}^{opt}\|$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$
V1	129	115	0	0	ML	0.298	6	3	0	ML	0.127	13	2	2	TL	0.116	116	13	126	2712	**0.077**
V2	157	135	0	1	ML	0.233	6	3	5	TL	0.091	21	5	4	ML	0.050	138	4	74	2535	**0.028**
V3	150	135	0	1	ML	0.224	6	4	3	ML	0.091	15	5	0	ML	0.080	137	12	148	3381	**0.038**
V4	150	135	0	0	ML	0.216	6	4	0	ML	0.105	16	4	1	ML	0.099	136	13	115	2845	**0.056**
V5	148	134	0	0	ML	0.228	5	2	0	ML	0.132	20	5	3	ML	0.053	136	12	141	2912	**0.031**
V6	149	135	0	0	ML	0.221	6	3	0	ML	0.111	16	2	1	TL	0.097	134	14	135	2638	**0.066**
V7	161	135	0	1	ML	0.222	6	4	2	TL	0.108	25	6	10	ML	0.033	138	1	30	1916	**0.023**
V8	170	156	0	0	ML	0.158	8	4	0	ML	0.083	16	2	0	TL	0.092	155	15	105	2058	**0.048**
V9	203	186	0	1	ML	0.111	7	5	3	ML	0.048	32	8	4	TL	0.027	187	15	102	2584	**0.012**
V10	200	186	0	1	ML	0.113	7	3	3	ML	0.066	25	9	2	ML	0.033	188	12	108	2052	**0.015**
V11	200	186	0	0	ML	0.104	7	5	0	ML	0.053	36	5	1	ML	0.046	186	12	45	1186	**0.021**
V12	199	186	0	0	ML	0.104	7	4	1	ML	0.054	19	7	0	ML	0.041	186	13	67	1485	**0.017**
V13	258	242	0	0	ML	0.003	8	4	0	ML	0.039	22	5	1	TL	0.034	184	73	10	1334	**0.001**
V14	219	186	0	0	ML	0.109	31	17	1	ML	0.015	80	13	6	ML	0.012	189	15	22	1569	**0.007**
V15	249	238	0	0	ML	0.001	10	5	0	TL	0.034	18	6	0	TL	0.025	204	44	3	811	**0.000**
V16	**244**	244	0	0	78	**opt**	244	0	0	411	**opt**	244	0	0	535	**opt**	233	11	0	346	**opt**
V17	**244**	244	0	0	61	**opt**	244	0	0	300	**opt**	244	0	0	419	**opt**	233	11	0	273	**opt**
V18	**244**	244	0	0	183	**opt**	76	37	0	TL	0.002	244	0	0	714	**opt**	233	11	0	286	**opt**
V19	**244**	244	0	0	55	**opt**	244	0	0	283	**opt**	244	0	0	298	**opt**	233	11	0	259	**opt**
V20	**243**	243	0	0	61	**opt**	44	27	0	TL	0.005	141	11	0	TL	0.004	232	11	0	270	**opt**
V21	**244**	244	0	0	110	**opt**	244	0	0	621	**opt**	244	0	0	567	**opt**	233	11	0	550	**opt**
V22	**239**	239	0	0	40	**opt**	239	0	0	181	**opt**	239	0	0	133	**opt**	239	0	0	189	**opt**
V23	225	153	0	0	ML	0.653	58	24	45	TL	**0.021**	15	6	5	TL	0.067	30	35	220	TL	0.021
V24	**239**	239	0	0	55	**opt**	239	0	0	267	**opt**	239	0	0	192	**opt**	239	0	0	283	**opt**
V25	**239**	239	0	0	44	**opt**	239	0	0	219	**opt**	239	0	0	155	**opt**	239	0	0	225	**opt**
V26	**239**	239	0	0	39	**opt**	239	0	0	200	**opt**	239	0	0	148	**opt**	239	0	0	207	**opt**
V27	**239**	239	0	0	40	**opt**	239	0	0	197	**opt**	239	0	0	144	**opt**	239	0	0	199	**opt**
V28	**239**	239	0	0	40	**opt**	239	0	0	180	**opt**	239	0	0	133	**opt**	239	0	0	191	**opt**

Table 5: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\|\mathcal{Z}^{opt}\|$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$
V29	**239**	239	0	0	78	**opt**	239	0	0	386	**opt**	239	0	0	245	**opt**	239	0	0	379	**opt**
V30	**238**	238	0	0	49	**opt**	238	0	0	254	**opt**	238	0	0	182	**opt**	238	0	0	263	**opt**
V31	**238**	238	0	0	41	**opt**	238	0	0	213	**opt**	238	0	0	155	**opt**	238	0	0	221	**opt**
V32	**238**	238	0	0	36	**opt**	238	0	0	191	**opt**	238	0	0	143	**opt**	238	0	0	200	**opt**
V33	**238**	238	0	0	35	**opt**	238	0	0	188	**opt**	238	0	0	141	**opt**	238	0	0	191	**opt**
V34	**238**	238	0	0	38	**opt**	238	0	0	167	**opt**	238	0	0	128	**opt**	238	0	0	182	**opt**
V35	**238**	238	0	0	69	**opt**	238	0	0	347	**opt**	238	0	0	234	**opt**	238	0	0	357	**opt**
V36	77	52	0	1	ML	0.850	3	0	0	ML	0.470	14	4	11	ML	0.065	21	15	134	TL	**0.039**
V37	96	57	0	1	ML	0.867	6	2	15	TL	0.126	13	2	13	TL	0.062	23	11	49	TL	**0.042**
V38	82	54	0	0	ML	0.848	6	5	3	TL	0.093	15	4	14	ML	0.050	31	16	224	TL	**0.027**
V39	70	31	0	0	ML	0.870	4	2	0	ML	0.192	8	2	5	ML	0.094	49	13	193	TL	**0.056**
V40	56	32	0	1	ML	0.858	4	2	0	ML	0.202	16	5	5	TL	0.064	41	13	210	TL	**0.037**
V41	86	42	0	1	ML	0.766	3	0	0	ML	0.455	24	5	14	ML	0.051	51	29	108	TL	**0.037**
V42	80	42	0	0	ML	0.776	5	5	3	ML	0.156	10	3	6	ML	0.122	54	24	167	TL	**0.054**
V43	67	42	0	0	ML	0.777	6	3	1	ML	0.126	14	4	7	ML	0.062	54	10	192	TL	**0.037**
V44	71	41	0	0	ML	0.771	6	4	2	ML	0.101	19	5	4	ML	0.073	54	15	219	TL	**0.043**
V45	92	41	0	0	ML	0.764	3	0	0	ML	0.456	25	6	10	TL	0.063	47	20	61	TL	**0.038**
V46	93	66	0	1	ML	0.559	8	5	9	TL	0.075	14	4	2	ML	0.091	71	13	139	TL	**0.043**
V47	114	89	0	0	ML	0.378	6	6	3	TL	0.112	22	2	3	TL	0.097	86	24	145	TL	**0.056**
V48	108	92	0	1	ML	0.399	6	4	2	ML	0.094	13	3	1	TL	0.098	89	18	167	TL	**0.057**
V49	599	395	0	0	TL	0.171	306	130	44	TL	0.001	334	81	51	TL	**0.001**	57	206	84	TL	0.007
V50	119	102	0	1	ML	0.330	7	4	6	ML	0.073	16	3	7	ML	0.052	99	0	53	1624	**0.039**
V51	116	102	0	1	ML	0.340	7	5	4	TL	0.090	13	2	3	ML	0.092	98	14	128	3220	**0.047**
V52	136	110	0	0	ML	0.329	7	4	3	ML	0.088	23	2	9	ML	0.084	99	22	113	2878	**0.055**
V53	128	113	0	0	ML	0.316	6	4	1	ML	0.107	15	2	1	TL	0.128	98	13	160	3374	**0.067**
V54	114	97	0	0	ML	0.335	7	3	7	ML	0.086	16	3	8	ML	0.053	99	2	43	2545	**0.037**

Table 6: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\|\mathcal{Z}^{opt}\|$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$
A1	613	492	0	0	ML	0.133	187	103	64	TL	**0.003**	45	38	28	TL	0.018	45	224	122	TL	0.008
A2	100	59	0	0	ML	0.779	30	10	23	TL	0.034	17	3	17	TL	0.047	63	36	503	3022	**0.019**
A3	100	72	0	0	ML	0.709	10	7	7	TL	0.063	16	6	9	ML	0.051	83	17	483	2908	**0.020**
A4	191	74	0	0	ML	0.702	58	17	39	TL	0.020	16	5	21	TL	0.037	107	31	155	TL	**0.019**
A5	376	332	0	0	ML	0.097	120	41	21	ML	0.010	13	7	1	ML	0.046	165	205	108	795	**0.008**
A6	**478**	387	0	0	ML	0.047	359	106	0	TL	0.000	50	9	0	TL	0.032	165	313	0	497	**opt**
A7	**446**	446	0	0	1568	**opt**	173	82	1	ML	0.001	10	5	1	ML	0.062	190	255	0	932	0.000
A8	290	247	0	0	ML	0.119	127	37	19	ML	0.008	10	5	1	TL	0.065	248	42	114	674	**0.006**
A9	**404**	404	0	0	2245	**opt**	123	53	0	ML	0.002	10	6	0	ML	0.042	249	155	0	578	**opt**
A10	**404**	404	0	0	137	**opt**	404	0	0	2448	**opt**	103	9	1	TL	0.016	251	152	1	403	0.000
A11	**404**	404	0	0	2522	**opt**	309	88	0	ML	0.000	93	8	1	ML	0.020	249	155	0	427	**opt**
A12	308	263	0	0	ML	0.102	75	33	14	ML	0.013	10	7	0	TL	0.078	250	52	91	3178	**0.007**
A13	**352**	329	0	0	ML	0.001	352	0	0	1920	**opt**	148	21	0	ML	0.015	326	26	0	292	**opt**
A14	**352**	352	0	0	56	**opt**	325	27	0	2228	**opt**	126	12	0	ML	0.017	326	26	0	283	**opt**
A15	**348**	348	0	0	1014	**opt**	348	0	0	359	**opt**	194	21	0	ML	0.006	338	10	0	300	**opt**
A16	**345**	345	0	0	81	**opt**	345	0	0	431	**opt**	345	0	0	303	**opt**	345	0	0	441	**opt**
L1	93	45	0	0	ML	0.933	6	2	3	TL	0.204	16	3	35	TL	**0.035**	34	13	56	2311	0.648
L2	158	67	0	0	ML	0.897	5	2	4	TL	0.206	7	3	27	TL	**0.076**	120	3	16	TL	0.256
L3	248	131	0	1	ML	0.760	17	6	27	TL	0.039	21	6	33	TL	0.023	78	6	50	TL	**0.019**
L4	341	235	0	1	ML	0.576	4	0	2	TL	0.193	14	7	22	ML	0.028	137	14	32	TL	**0.024**
L5	290	245	0	1	ML	0.542	6	2	9	TL	0.115	17	4	13	TL	0.040	233	0	36	TL	**0.038**
L6	331	241	0	0	ML	0.555	15	5	12	ML	0.051	15	7	18	TL	0.031	132	11	35	TL	**0.026**
L7	473	233	0	0	ML	0.523	23	14	18	ML	0.026	21	9	13	TL	0.035	406	4	34	TL	**0.020**
L8	363	345	0	0	ML	0.349	7	2	3	TL	0.128	12	5	7	TL	**0.072**	236	11	1	TL	0.206
L9	388	357	0	1	ML	0.339	11	5	9	ML	0.057	13	5	13	ML	0.042	339	0	27	TL	**0.029**
L10	430	295	0	0	ML	0.381	6	1	2	ML	0.184	50	10	16	TL	0.026	213	20	15	TL	**0.019**
L11	554	280	0	0	ML	0.400	5	3	0	ML	0.108	68	11	15	TL	0.026	423	24	19	TL	**0.019**
L12	609	446	0	2	ML	0.293	5	2	0	ML	0.133	58	4	8	TL	0.059	511	1	17	TL	**0.050**

Table 7: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOTAConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\lvert\mathcal{Z}^{opt}\rvert$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$
T1	**241**	241	0	0	884	**opt**	229	9	0	TL	0.000	241	0	0	2353	**opt**	37	175	17	TL	0.000
T2	**228**	228	0	0	1005	**opt**	183	29	7	TL	0.000	228	0	0	2157	**opt**	34	154	10	2916	0.000
T3	**230**	230	0	0	2287	**opt**	119	52	20	TL	0.001	229	1	0	TL	**opt**	33	156	21	TL	0.000
T4	**242**	242	0	0	902	**opt**	197	26	6	TL	0.000	242	0	0	2145	**opt**	32	187	14	TL	0.000
T5	117	13	0	5	ML	0.437	40	25	26	TL	0.016	32	20	18	TL	0.012	35	41	83	TL	**0.008**
T6	**233**	233	0	0	705	**opt**	233	0	0	TL	**opt**	233	0	0	1807	**opt**	33	164	20	2564	0.000
T7	**189**	189	0	0	1067	**opt**	180	8	1	TL	0.000	189	0	0	2287	**opt**	29	130	14	2495	0.000
T8	**223**	223	0	0	270	**opt**	223	0	0	1369	**opt**	223	0	0	757	**opt**	30	154	24	1309	0.000
T9	**239**	239	0	0	939	**opt**	239	0	0	2702	**opt**	239	0	0	1238	**opt**	28	188	16	2093	0.000
T10	**221**	221	0	0	592	**opt**	221	0	0	2244	**opt**	221	0	0	978	**opt**	28	155	26	1770	0.000
T11	**227**	227	0	0	399	**opt**	227	0	0	1684	**opt**	227	0	0	901	**opt**	27	172	19	1489	0.000
T12	**222**	222	0	0	310	**opt**	222	0	0	1638	**opt**	222	0	0	877	**opt**	28	161	19	1683	0.000
T13	**240**	240	0	0	416	**opt**	240	0	0	1698	**opt**	240	0	0	925	**opt**	28	195	12	1677	0.000
T14	**249**	249	0	0	612	**opt**	249	0	0	2211	**opt**	249	0	0	1117	**opt**	28	186	23	2150	0.000
T15	**227**	227	0	0	402	**opt**	227	0	0	1680	**opt**	227	0	0	927	**opt**	27	165	23	1645	0.000
T16	**223**	223	0	0	393	**opt**	223	0	0	1989	**opt**	223	0	0	998	**opt**	28	167	15	1626	0.000
T17	**241**	241	0	0	413	**opt**	241	0	0	1586	**opt**	241	0	0	969	**opt**	27	173	26	1595	0.000
T18	**218**	218	0	0	202	**opt**	218	0	0	1627	**opt**	218	0	0	537	**opt**	39	179	0	1979	**opt**
T19	**244**	244	0	0	486	**opt**	244	0	0	1810	**opt**	244	0	0	952	**opt**	25	186	24	1994	0.000
R1	**353**	353	0	0	298	**opt**	52	44	19	TL	0.007	353	0	0	638	**opt**	72	281	0	3347	**opt**
R2	**368**	368	0	0	373	**opt**	368	0	0	2168	**opt**	368	0	0	695	**opt**	73	285	7	2100	0.000
R3	**410**	410	0	0	748	**opt**	395	13	1	TL	0.000	410	0	0	1448	**opt**	73	317	19	2634	0.000
R4	**367**	367	0	0	328	**opt**	367	0	0	TL	**opt**	367	0	0	814	**opt**	79	273	12	2223	0.000
R5	**399**	399	0	0	1808	**opt**	157	71	43	TL	0.001	399	0	0	2748	**opt**	65	261	40	TL	0.001
R6	**448**	448	0	0	151	**opt**	448	0	0	867	**opt**	448	0	0	393	**opt**	66	368	10	1043	0.000
R7	**447**	447	0	0	177	**opt**	447	0	0	827	**opt**	447	0	0	404	**opt**	60	361	23	989	0.000
R8	**405**	405	0	0	340	**opt**	405	0	0	1742	**opt**	405	0	0	746	**opt**	56	336	11	1569	0.000
R9	**405**	405	0	0	243	**opt**	405	0	0	1166	**opt**	405	0	0	517	**opt**	58	339	8	1190	0.000
R10	**419**	419	0	0	246	**opt**	419	0	0	1183	**opt**	419	0	0	573	**opt**	64	335	14	1161	0.000
R11	**449**	449	0	0	236	**opt**	449	0	0	1100	**opt**	449	0	0	525	**opt**	60	352	37	1048	0.000
R12	**409**	409	0	0	1177	**opt**	409	0	0	2844	**opt**	409	0	0	1306	**opt**	60	319	22	1921	0.000
R13	**322**	322	0	0	136	**opt**	4	1	1	ML	0.243	322	0	0	538	**opt**	87	225	10	TL	0.000
R14	**446**	446	0	0	351	**opt**	446	0	0	1258	**opt**	446	0	0	663	**opt**	49	380	16	1496	0.000

Table 8: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOTAConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\|\mathcal{Z}^{opt}\|$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$
V1	680	163	1	0	ML	0.694	32	40	26	TL	0.016	311	99	110	TL	**0.001**	116	216	128	TL	0.011
V2	389	51	0	2	ML	0.632	18	14	29	TL	0.030	235	34	157	TL	**0.002**	98	37	89	TL	0.010
V3	616	219	0	4	TL	0.428	68	68	45	TL	0.007	404	37	100	TL	**0.001**	99	193	112	TL	0.011
V4	805	593	0	1	TL	0.109	15	18	13	TL	0.037	522	80	51	TL	**0.000**	101	253	94	TL	0.007
V5	781	535	2	3	TL	0.157	114	111	58	TL	0.004	452	92	76	TL	**0.001**	100	233	131	TL	0.007
V6	646	130	1	7	ML	0.507	36	36	40	TL	0.014	274	102	105	TL	**0.001**	113	206	146	TL	0.008
V7	189	13	0	2	ML	0.954	10	6	10	TL	0.069	78	18	74	TL	**0.011**	91	13	120	TL	0.016
V8	638	146	0	3	ML	0.674	57	59	35	TL	0.009	267	91	113	TL	**0.001**	113	203	146	TL	0.008
V9	442	33	0	0	ML	0.919	18	13	29	TL	0.031	287	28	173	TL	**0.001**	98	60	151	TL	0.011
V10	581	58	0	1	ML	0.743	50	53	44	TL	0.010	395	46	121	TL	**0.001**	99	149	168	TL	0.010
V11	662	105	0	5	ML	0.454	66	68	48	TL	0.007	499	46	94	TL	**0.001**	100	187	156	TL	0.010
V12	790	635	0	0	TL	0.072	163	141	63	TL	0.002	499	92	40	TL	**0.000**	96	224	92	TL	0.009
V13	**801**	801	0	0	1614	**opt**	112	105	67	TL	0.003	801	0	0	3523	**opt**	96	350	64	TL	0.008
V14	173	12	0	2	ML	0.964	9	5	5	TL	0.072	70	20	61	TL	**0.011**	95	5	93	TL	0.017
V15	**801**	801	0	0	1947	**opt**	170	135	72	TL	0.002	774	20	1	TL	0.000	95	345	75	TL	0.008
V16	676	393	0	2	TL	0.204	63	62	38	TL	0.008	367	80	75	TL	**0.001**	93	208	94	TL	0.011
V17	**766**	766	0	0	3515	**opt**	131	118	63	TL	0.003	530	52	38	TL	0.000	91	270	90	TL	0.009
V18	643	27	0	1	ML	0.857	38	41	36	TL	0.016	475	49	99	TL	**0.001**	97	176	147	TL	0.009
V19	**735**	735	0	0	2543	**opt**	62	68	34	TL	0.008	573	48	40	TL	0.000	93	291	100	TL	0.006
V20	**803**	803	0	0	1934	**opt**	247	173	78	TL	0.001	688	54	9	TL	0.000	93	324	99	TL	0.007
V21	445	97	0	4	TL	0.484	74	61	46	TL	0.008	213	63	117	TL	**0.002**	93	93	178	TL	0.013
V22	803	75	0	4	ML	0.620	83	83	45	TL	0.006	633	70	24	TL	**0.000**	93	273	84	TL	0.007
V23	685	378	0	6	TL	0.263	44	45	39	TL	0.011	374	72	165	TL	**0.001**	155	82	143	TL	0.004
V24	561	79	1	2	TL	0.609	105	91	58	TL	0.004	242	79	97	TL	**0.002**	84	176	125	TL	0.028
V25	586	72	1	0	ML	0.789	120	105	59	TL	0.004	304	86	114	TL	**0.001**	84	193	105	TL	0.029
V26	707	391	0	0	TL	0.221	143	121	60	TL	0.003	393	90	67	TL	**0.001**	88	238	99	TL	0.012
V27	720	483	0	1	TL	0.105	118	104	51	TL	0.003	450	82	54	TL	**0.000**	88	245	90	TL	0.017
V28	803	75	0	3	ML	0.620	109	106	53	TL	0.003	642	72	23	TL	**0.000**	93	251	81	TL	0.008

Table 9: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOTAConFL.
$|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\lvert\mathcal{Z}^{opt}\rvert$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$	$\lvert\mathcal{Z}^{*}\rvert$	$\lvert\mathcal{Z}^{+}\rvert$	$\lvert\mathcal{Z}^{-}\rvert$	$t[s]$	$gH[\%]$
V29	386	41	0	1	ML	0.869	120	88	84	TL	**0.005**	164	45	88	TL	0.005	84	68	210	TL	0.027
V30	431	48	0	2	ML	0.677	127	92	79	TL	**0.004**	178	39	62	TL	0.007	82	97	204	TL	0.031
V31	407	29	0	1	ML	0.682	118	80	81	TL	**0.005**	196	43	75	TL	0.005	82	97	204	TL	0.031
V32	347	16	0	2	ML	0.807	114	77	78	TL	**0.005**	191	32	85	TL	0.006	82	79	238	TL	0.023
V33	317	10	0	2	ML	0.974	84	63	69	TL	**0.008**	148	30	51	TL	0.011	82	69	242	TL	0.021
V34	318	13	0	0	ML	0.971	96	74	80	TL	**0.006**	120	24	37	TL	0.016	82	65	253	TL	0.030
V35	381	45	0	6	ML	0.648	125	74	87	TL	**0.005**	142	33	53	TL	0.010	82	65	198	TL	0.034
V36	547	232	0	1	TL	0.595	23	25	36	TL	0.020	245	72	133	TL	**0.002**	140	50	107	TL	0.005
V37	363	72	0	3	ML	0.595	11	10	15	TL	0.053	144	46	101	TL	**0.003**	147	23	94	TL	0.007
V38	**801**	801	0	0	2776	**opt**	24	28	21	TL	0.021	468	91	47	TL	0.000	134	188	100	TL	0.005
V39	687	109	0	2	TL	0.709	12	12	12	ML	0.047	597	28	104	TL	**0.000**	140	277	130	TL	0.005
V40	**869**	869	0	0	1888	**opt**	101	99	54	TL	0.003	591	119	51	TL	0.000	136	251	133	TL	0.004
V41	411	109	0	4	ML	0.601	56	44	65	TL	0.009	186	45	132	TL	**0.002**	124	30	110	TL	0.007
V42	544	188	0	1	TL	0.545	83	69	91	TL	0.005	290	73	107	TL	**0.001**	139	52	98	TL	0.006
V43	796	503	0	2	ML	0.232	12	14	12	TL	0.044	556	87	62	TL	**0.001**	141	194	110	TL	0.005
V44	**837**	837	0	0	1579	**opt**	105	88	59	TL	0.004	616	83	48	TL	0.000	132	305	142	TL	0.004
V45	264	36	0	3	ML	0.676	13	11	13	TL	0.049	96	42	84	TL	**0.005**	110	17	112	TL	0.009
V46	565	209	1	3	TL	0.571	61	56	65	TL	0.008	299	63	128	TL	**0.001**	135	54	78	TL	0.006
V47	**832**	832	0	0	3261	**opt**	112	97	61	TL	0.003	582	67	47	TL	0.000	119	254	108	TL	0.005
V48	731	418	1	1	ML	0.285	13	14	11	TL	0.037	507	61	91	TL	**0.001**	130	208	145	TL	0.006
V49	**851**	851	0	0	1341	**opt**	7	7	9	ML	0.088	514	84	95	TL	0.000	198	109	128	TL	0.001
V50	321	32	1	1	ML	0.919	19	18	25	TL	0.028	180	41	135	TL	**0.002**	104	29	105	TL	0.010
V51	508	173	0	5	TL	0.522	32	33	28	TL	0.016	293	58	129	TL	**0.001**	115	104	166	TL	0.008
V52	766	413	0	1	TL	0.293	87	85	56	TL	0.005	460	83	65	TL	**0.001**	118	164	151	TL	0.006
V53	**834**	834	0	0	3055	**opt**	114	105	60	TL	0.003	509	83	63	TL	0.000	111	272	107	TL	0.005
V54	162	11	0	1	ML	0.969	9	6	15	TL	0.069	49	24	55	TL	**0.014**	104	6	94	TL	0.017

Table 10: Overview of runtime and size of Pareto front for the methods *eps*, *rectB*, *asosB* and *ilph* for BOTAConFL. $|\mathcal{Z}^{opt}|$: size of Pareto front, bold if complete front found; $|\mathcal{Z}^{*}|$: Pareto optimal points discovered and proven; $|\mathcal{Z}^{+}|$: Pareto optimal points discovered, but not proven; $|\mathcal{Z}^{-}|$: further points discovered; $t[s]$: runtime; $gH[\%]$: hypervolume gap (wrt. to best rH of all methods).

		eps					*rectB*					*asosB*					*ilph*				
	$\|\mathcal{Z}^{opt}\|$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$	$\|\mathcal{Z}^{*}\|$	$\|\mathcal{Z}^{+}\|$	$\|\mathcal{Z}^{-}\|$	$t[s]$	$gH[\%]$
A1	463	75	1	3	ML	0.514	30	26	25	TL	0.018	288	85	112	TL	**0.001**	211	76	146	TL	0.002
A2	310	60	0	1	ML	0.579	3	1	2	TL	0.403	86	59	44	TL	0.006	207	35	214	TL	**0.004**
A3	367	77	0	2	ML	0.540	25	23	24	ML	0.023	140	71	72	TL	0.004	200	96	200	TL	**0.003**
A4	349	58	0	4	ML	0.527	8	2	18	TL	0.087	113	58	67	TL	**0.005**	191	23	105	TL	0.005
A5	293	42	1	1	ML	0.554	52	37	30	TL	0.014	65	41	26	TL	0.010	184	76	240	TL	**0.004**
A6	287	66	1	1	ML	0.543	13	8	15	TL	0.051	59	42	27	TL	0.012	187	74	303	TL	**0.002**
A7	258	58	0	2	ML	0.567	8	3	8	TL	0.089	51	31	35	TL	0.013	177	49	274	TL	**0.005**
A8	271	53	0	4	ML	0.545	59	45	32	TL	0.012	44	30	29	TL	0.015	171	76	261	TL	**0.005**
A9	266	49	1	1	ML	0.553	26	21	16	TL	0.027	49	34	29	TL	0.015	179	69	279	TL	**0.003**
A10	293	46	0	3	ML	0.524	58	40	27	TL	0.012	82	40	39	TL	0.010	167	92	251	TL	**0.004**
A11	429	59	0	3	ML	0.640	108	78	48	TL	0.005	155	82	66	TL	0.004	179	168	197	TL	**0.003**
A12	420	102	1	2	ML	0.481	109	84	52	TL	**0.005**	31	23	14	TL	0.024	161	139	180	TL	0.006
A13	406	111	0	2	ML	0.423	212	92	110	TL	**0.004**	51	32	24	TL	0.020	154	154	233	TL	0.005
A14	405	108	1	2	ML	0.372	216	89	116	TL	**0.004**	53	36	24	TL	0.018	154	177	218	TL	0.005
A15	320	68	1	3	ML	0.497	8	6	8	TL	0.081	117	52	62	TL	0.007	174	98	276	TL	**0.003**
A16	410	120	1	3	ML	0.561	218	90	119	TL	**0.005**	31	20	16	TL	0.030	152	148	232	TL	0.006
L1	222	9	0	2	ML	0.986	8	0	11	TL	0.088	14	5	33	TL	**0.035**	179	4	231	TL	0.193
L2	344	23	0	0	ML	0.939	10	2	15	TL	0.075	64	50	73	TL	**0.006**	113	19	206	TL	0.006
L3	304	20	0	2	ML	0.954	8	3	10	TL	0.093	89	57	88	TL	**0.005**	141	24	210	TL	0.030
L4	377	8	0	1	ML	0.977	8	1	9	TL	0.111	40	28	59	TL	**0.009**	132	61	179	TL	0.022
L5	380	23	0	0	ML	0.954	18	4	39	TL	0.036	88	48	106	TL	0.005	123	22	221	TL	**0.004**
L6	311	4	0	1	ML	0.997	13	5	23	TL	0.067	33	20	45	TL	**0.014**	105	8	177	TL	0.038
L7	428	13	0	0	ML	0.971	13	2	24	TL	0.069	82	45	77	TL	**0.007**	130	63	179	TL	0.008
L8	477	12	0	2	ML	0.969	31	19	45	TL	0.021	95	55	100	TL	0.005	155	50	234	TL	**0.004**
L9	527	19	0	2	ML	0.962	41	18	70	TL	0.015	120	60	90	TL	0.005	172	55	239	TL	**0.004**

6 Conclusions

In this paper, we introduce a new exact method, two matheuristics and, to the best of our knowledge, a first two-phase ILP-based heuristic approach for bi-objective binary problems. The exact method is a combination of the well-known ϵ-constraint method (5; 14) and the binary search in objective space (7; 13). The two matheuristics, BINS and directional local branching, are bi-objective counterparts of two matheuristics that are known to work well in single-objective context. Both matheuristics are used within exact frameworks to generate solutions that may be Pareto optimal. They are also main ingredients of our two-phase ILP-based heuristic.

The computational experiments show that our exact method outperforms other methods from literature and the proposed matheuristics are not only a useful support for exact methods, but also perform quite well when used within a two-phase ILP-based heuristic solution framework. Since both the exact method and the heuristics can be easily implemented using commercial ILP-solvers, our hope – in the same spirit as (6) – is that practitioners will be encouraged to use ILP-methods for solving bi-objective integer problems. Last but not least, we believe that this study will motivate further research on the boundary area between mixed integer programming and metaheuristics for bi/multi-objective optimization.

Acknowledgement

Markus Leitner is supported by the Austrian Science Fund (FWF) under grant I892-N23. Markus Sinnl is supported by the Austrian Science Fund (FWF) under grant P26755-N19. Axel Werner is supported by the German Research Foundation (DFG) under grant GR 883/18-1.

References

[1] P. Belotti, B. Soylu, M. M. Wiecek, A branch-and-bound algorithm for biobjective mixed-integer programs, Optimization Online.

[2] N. Jozefowiez, G. Laporte, F. Semet, A generic branch-and-cut algorithm for multiobjective optimization problems: Application to the multilabel traveling salesman problem, INFORMS J. on Comp. 24 (1) (2012) 554–564.

[3] S. N. Parragh, F. Tricoire, Branch-and-bound for bi-objective integer programming, Optimization Online.

[4] T. Stidsen, K. A. Andersen, B. Dammann, A Branch and Bound Algorithm for a Class of Biobjective Mixed Integer Programs, Management Sci. 60 (4) (2014) 1009–1032.

[5] J. F. Bérubé, M. Gendreau, J. Y. Potvin, An exact ϵ-constraint method for bi-objective combinatorial optimization problems: Application to the Traveling Salesman problem with Profits, Eur. J. Oper. Res. 194 (1) (2009) 39–50.

[6] N. Boland, H. Charkhgard, M. Savelsbergh, A Criterion Space Search Algorithm for Biobjective Mixed Integer Programming: The Rectangle Splitting Method, Optimization Online.

[7] J. Riera-Ledesma, J. J. Salazar-González, The biobjective travelling purchaser problem, Eur. J. Oper. Res. 160 (3) (2005) 599–613.

[8] C. A. C. Coello, A comprehensive survey of evolutionary-based multiobjective optimization techniques, Knowledge and Information systems 1 (3) (1999) 269–308.

[9] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation 1 (1) (2011) 32–49.

[10] E. Danna, E. Rothberg, C. L. Pape, Exploring relaxation induced neighborhoods to improve MIP solutions, Math. Programming 102 (1) (2005) 71–90.

[11] M. Fischetti, A. Lodi, Local branching, Math. Programming 98 (1-3) (2003) 23–47.

[12] M. Fischetti, M. Monaci, Proximity search for 0-1 mixed-integer convex programming, J. of Heuristics 20 (6) (2014) 709–731.

[13] L. G. Chalmet, L. Lemonidis, D. J. Elzinga, An algorithm for the bi-criterion integer programming problem, Eur. J. Oper. Res. 25 (2) (1986) 292–300.

[14] V. Chankong, Y. Y. Haimes, Multiobjective decision making: theory and methodology, no. 8, North-Holland, 1983.

[15] M. Fischetti, M. Leitner, I. Ljubic, M. Luipersbeck, M. Monaci, M. Resch, D. Salvagnin, M. Sinnl, Thinning out Steiner trees: a node-based model for uniform edge costs, Technical Report.

[16] T. Koch, A. Martin, Solving Steiner tree problems in Graphs to Optimality, Networks 32 (1998) 207–232.

[17] S. Gollowitzer, I. Ljubić, MIP models for connected facility location: A theoretical and computational study, Comput. & Oper. Res. 38 (2) (2011) 435–449.

[18] M. Leitner, G. R. Raidl, Branch-and-cut-and-price for capacitated connected facility location, J. of Math. Modelling and Algorithms 10 (3) (2011) 245–267.

[19] A. Przybylski, X. Gandibleux, M. Ehrgott, Two phase algorithms for the bi-objective assignment problem, Eur. J. Oper. Res. 185 (2) (2008) 509–533.

[20] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, V. G. Da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, Evolutionary Computation, IEEE Transactions on 7 (2) (2003) 117–132.

[21] E. Ulungu, J. Teghem, The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems, Foundations of Comput. and Decision Sci. 20 (2) (1995) 149–165.

[22] M. Leitner, I. Ljubić, M. Sinnl, A Computational Study of Exact Approaches for the Bi-Objective Prize-Collecting Steiner Tree Problem, INFORMS J. Comput. 27 (1) (2015) 118–134.

[23] F. Tricoire, Multi-directional local search, Comput. & Oper. Res. 39 (12) (2012) 3089–3101.

[24] T. Lust, J. Teghem, Two-phase Pareto local search for the biobjective traveling salesman problem, J. of Heuristics 16 (3) (2010) 475–510.

[25] M. Grötschel, C. Raack, A. Werner, Towards optimizing the deployment of optical access networks, EURO J. on Comput. Optim. 2 (1-2) (2014) 17–53.

[26] M. Leitner, I. Ljubić, M. Sinnl, A. Werner, On the two-architecture connected facility location problem, Electronic Notes in Discrete Mathematics 41 (2013) 359–366.

[27] I. Ljubić, R. Weiskircher, U. Pferschy, G. W. Klau, P. Mutzel, M. Fischetti, An Algorithmic Framework for the Exact Solution of the Prize-Collecting Steiner tree problem, Math. Programming 105 (2006) 427–449.

[28] B. V. Cherkassky, A. V. Goldberg, On implementing the push - relabel method for the maximum flow problem, Algorithmica (1997) 390–410.