universität wien

# DISSERTATION

Titel der Dissertation

Matheuristic Algorithms for Solving Multi-objective/Stochastic Scheduling and Routing Problems

Verfasser

Mag. Peter Reiter Bakk.

angestrebter akademischer Grad

Doktor der technischen Wissenschaften (Dr. techn.)

Wien, 2010

| | |
|---|---|
| Studienkennzahl lt. Studienblatt: | A 786 175 |
| Studienrichtung lt. Studienblatt: | Wirtschaftsinformatik |
| Betreuer: | Ao. Univ.-Prof. Mag. Dr. Walter J. Gutjahr |

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# 1. Introduction

Optimization problems appear in many practically relevant areas of our life. Typical application areas are project scheduling and staffing, production planning, transportation, investment planning and many more. Improvements in solution often have a direct impact on costs and other important factors like customer satisfaction. It is well known that only special classes of optimization problems (like linear optimization problems) can be efficiently solved by polynomial time algorithms. Many real world problems are hard to solve due to additional requirements e.g. the problem may have a combinatorial structure, non-linearities are present or uncertainty needs to be considered. Considering complex real-world applications the list of difficulties can be arbitrarily extended and each application is in some way unique. In this work we consider in general computationally difficult *combinatorial optimization problems* (COPs). To solve such problems a large number of algorithmic solution approaches exist. These approaches can be classified into two main categories (i) *exact* and (ii) *heuristic* algorithms, each class having its assets and drawbacks. Exact approaches like *branch-and-bound*, *dynamic programming*, *constraint programming*, and the large class of *integer linear programming* techniques (e.g. branch-and-cut, branch-and-price, branch-and-cut-and-price (Nemhauser and Wolsey [92], Papadimitriou and Steiglitz [94])) are guaranteed to find an optimal solution and provide a guarantee that the solution is indeed optimal. In general the run-time often increases dramatically with instance sizes, therefore only small/moderately sized instances can be solved (within reasonable run-times).
On the other hand heuristic algorithms, that trade optimality for run-time are applicable to larger instances. Especially *metaheuristics* have proven to be highly useful in practice. This class includes, among others, variable neighborhood search, simulated annealing, various population based methods (e.g. evolutionary algorithms), and estimation of distribution algorithms (e.g. ant colony optimization) (Glover and Kochenberger [48], Hoos and Stützle [65]).

The assets and drawbacks of the two classes can be seen as complementary, therefore combining ideas from both streams appears to be natural. Hybrid algorithms combining elements of both streams, have proven to be more efficient in terms of run-time and/or solution quality. Such hybrid algorithms are called *matheuristics*. Various models of combinations exist, examples are given in (Dumitrescu and Stützle[38], Puchinger and Raidl[100], Raidl[101]) and a classification if provided in (Talbi[116]). In this work we will consider two different combinations (i) an exact algorithm is used to solve a subproblem within a heuristic framework (see Chapter 3) and (ii) heuristic algorithms are used to boost the performance of exact algorithms (see Chapter 4).

The first application that is considered in this thesis, the *Multi-objective Project Selection, Scheduling and Staffing with Learning* problem (short MPSSSL) (Chapter 3), arises from the field of management in research-centered organizations. Given a set of project proposals the decision makers have to select the "best" subset of projects (a project portfolio) and set these up properly (schedule them and provide the necessary resources). This problem is hard to solve for different reasons: (i) selecting a subset of projects considering limited resources is a knapsack-type problem that is known to be $\mathcal{NP}$-hard,
and (ii) to determine the feasibility of a given portfolio, the projects have to be scheduled and staff must be assigned to them. As in this problem the assignment of workers is influenced by the decision which portfolio should be selected, the decision maker has to consider goals of different nature. Some objectives are related to economic goals (e.g. return of investment), others are related to the competence development of the workers. Competence oriented goals are motivated by the fact that competencies determine the attainment and sustainability of strategic position in market competition. In general the objectives can not be combined to a single objective, therefore methods for solving multi-objective optimization problems are used. In practice uncertainty is another typically encountered aspect. Different parameters of the problem can be uncertain (e.g. benefits of a project, or the time and effort required to perform the single activities required by a project). To determine the "best" portfolio, methods are needed that are able to handle uncertainty in optimization. For abbreviation we refer to the stochastic extension as the SMPSSSL problem.

The second problem, the *Bi-objective Capacitated Vehicle Routing Problem with Route Balancing* (short CVRPB) (Chapter 4) arises from the field of vehicle routing. Given a set of customers, the decision makers have to construct routes for a fixed number of vehicles, each starting and ending at the same depot, such that the demands of all customers can be fulfilled, and the capacity constraints of each vehicle are not violated. The traditional objective of this problem (known as the Capacitated Vehicle Routing Problem (CVRP)[1]) is minimizing the total costs of all routes. A problem that may arise by this approach is that the resulting routes can be very unbalanced (in the sense of drivers workload). To

[1]As the CVRP is an extension of the well known Traveling Salesman Problem the CVRP is $\mathcal{NP}$-hard.

overcome this problem a second objective function that measures the balance of the routes of a solution is introduced.

Both application share the factor that multiple objectives are present. In the first application also uncertainty on different model parameters are considered. These are two common aspects that characterize many real-world problems. Although methodologies to treat these features exist, these issues are still in a developing phase. Especially the combined consideration of multi-objective and stochastic features in combinatorial optimization problems must be characterized as widely unexplored (cf. Fu[45]).

In this work different ways on how to solve the considered problems are presented. Various hybrid methods that combine exact, meta-heuristic and (in the stochastic case) simulation approaches have been developed.

Meta-heuristic methods capable to solve multi-objective combinatorial optimization problems based on the Nondominated Sorting Genetic Algorithm II (NSGA-II) by Deb et al.[33], and the Pareto Ant Colony (P-ACO) algorithm by Doerner et al.[35] combined with an linear programming solver as a subordinate have been implemented to treat the MPSSSL problem. To solve the stochastic extension SMPSSSL we implemented an algorithm that combines the aforementioned NSGA-II algorithm with a method by Gutjahr[54] that handles the interplay between multi-objective optimization and simulation called *Adaptive Pareto Sampling* (APS). APS uses a sampling approach for the estimation of expectations, that is based on *Monte-Carlo simulation*. To reduce the computational burden of the sampling approach without losing accuracy of the estimator we improve the simulation process by using *importance sampling* (IS) (see Rubinstein and Kroese[107]).

For the CVRPB problem, we use the adaptive $\varepsilon$-constraint method by Laumanns et al.[81] in combination with a branch-and-cut algorithm and two genetic algorithms (GAs), namely a single-objective GA and the multi-objective NSGA-II (Deb et al.[33]), to solve the considered problem. In general the adaptive $\varepsilon$-constraint method determines the Pareto set by solving a sequence of constrained single-objective problems. In our implementation this requires an efficient branch-and-cut algorithm capable of solving distance-constrained CVRP (DCVRP). Instead of a straightforward three-index problem formulation providing a special index for the vehicle under consideration, we apply a more efficient two-index formulation proposed by Laporte et al.[78,79] for the DCVRP. We have implemented different separation procedures to identify violated inequalities related to the distance constraints. These procedures require the computation of valid and efficient lower bounds for a multiple traveling salesman problem, therefore we apply generalized Held-Karp bounds for this purpose. To improve the performance of this exact approach, the GAs are applied to generate good incumbent candidates for the branch-and-cut algorithm in order to speed up the search process. They are called either in a sequential way (NSGA-II) or in an interactive way (single-objective GA).

The remainder of this thesis is organized as follows: In Chapter 2 an introduction to multi-objective optimization and simulation is provided. The used performance assessment methods and algorithms are briefly described. Chapter 3 presents the MPSSSL and SMPSSSL problem. A problem description, the corresponding mathematical models as well as a description of the problem specific parts of the algorithms are presented. At the end of Chapter 3, computational results and a problem specific conclusion are presented. In Chapter 4, the CVRPB problem and the corresponding solution procedures are stated, and computational results and conclusions are presented. Finally Chapter 5 provides a summary of the insights obtained during the development of the solution methods needed to solve the considered optimization problems.

This thesis is built on the results of two research projects funded by the Austrian Science Fund (FWF) grant L264-N13 and P20342, and related works Gutjahr and Reiter [55], Gutjahr et al. [57,58], Reiter and Gutjahr [105], Reiter et al. [106].

# 2. Basics

## 2.1. Multi-objective Optimization

This chapter is based on the introduction to multi-objective optimization given in Deb [29]. Multi-objective optimization problems (MOOP) deal with more than one objective function. Due to the lack of suitable solution methods in the past, a MOOP has been transformed and solved as a single objective problem. In single objective optimization the goal is to find one solution (or in special cases multiple optimal solutions). In multi-objective optimization it is not sufficient to find an optimal solution for each objective function. In the following, the general form of a MOOP is shown.

$$\begin{aligned}
\text{(MOOP) } \min/\max\ & f_k(x) && k = 1, 2, \ldots, t, \\
\text{s.t. } & g_j(x) \geq 0 && j = 1, 2, \ldots, \bar{m}, \\
& h_j(x) = 0 && j = \bar{m}+1, \ldots, m, \\
& x_i^L \leq x_i \leq x_i^U && i = 1, 2, \ldots, n.
\end{aligned} \tag{2.1}$$

A "solution" $x$ is a vector of $n$ decision variables: $x = (x_1, x_2, \ldots, x_n)^T$. The *decision variable space (decision space)* $X$ is bounded by the last set of constraints, where $x_i^L$ and $x_i^U$ are lower, upper bounds for the decision variables $x_i$. A solution that does not satisfy all constraints and variable bounds is called an *infeasible solution* all others are called *feasible solutions*. The set of all feasible solutions is called *feasible region* $\tilde{S} \subseteq X$.
A MOOP has $t$ objective functions $f(x) = (f_1(x), f_2(x), \ldots, f_t(x))^T$, each of them either can be minimized of maximized. According to the duality principle maximization problem can be transformed into minimization problems and vice versa. The biggest difference between single- and multi-objective optimizations is that in multi-objective optimization the objective functions constitute a multi-dimensional space (*objective space* $Z$). Each solution $x$ maps to a point $z$ in the objective space, where $f(x) = z = (z_1, z_2, \ldots, z_t)^T$. The mapping transfers an $n$-dimensional solution vector into an $t$-dimensional objective vector.

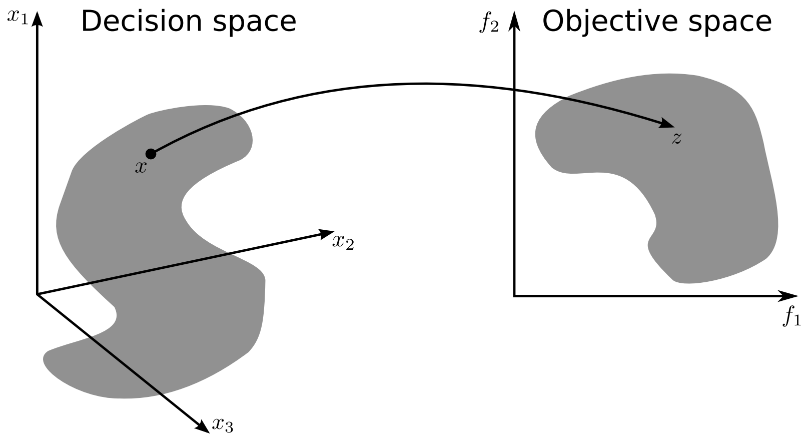



Figure 2.1.: Mapping between decision and objective space

### Classical Methods

According to Deb [29], *classical* methods are methods used to solve multi-objective optimization problems without the use of an evolutionary approach. Hwang and Masud [67] and Miettinen [90] suggested the following classification.

- No-preference methods: No information about the importance of objectives is used, but a heuristic approach to find a single optimal solution. These methods do not try to find multiple solutions, e.g. Method of Global Criterion, Multi-objective Proximal Bundle Method, etc.

- Posteriori methods: A set of Pareto-optimal solutions is generated by an algorithm and then presented to the decision maker who selects the most preferred solution, e.g. Method of Weighted Metrics (a special case is the Weighted Sum Method), $\epsilon$-Constraint Method, etc.

- A priori methods: A priori methods use information, decided in advance to generate solutions that correspond to the hopes of the decision maker, e.g. Value Function Method, Lexicographic Ordering, Goal Programming, etc.

- Interactive methods: The decision maker is part of the solution generating process. The preference structure of the decision maker is determined in an interactive way. The basic steps are: -find a feasible solution, interact with the decision maker, obtain a new solution. If a solution is accepted by the decision maker then the process stops, otherwise additional solutions are generated, e.g. Interactive Surrogate Worth Trade-Off Method, Geoffrion-Dyer-Feinberg Method, Chebyshev Method, etc.

For the remainder of this work we consider both classical and metaheuristic methods that rely on the concept of Pareto-optimal solutions.

### 2.1.1. Pareto-Optimal Solutions

When conflicting objectives are included in the MOOP so called *Pareto-optimal solutions* have to be found. As it can be seen in Figure 2.1 it is clear that not all solutions in the solution space $S$ are feasible. Solutions in the space $S$ can be mapped to a point in the objective space $Z$. So each point $x$ in the left graph corresponds to a point $z$ in the right graph. Therefore a comparison of any two solutions $x^{(1)}$ and $x^{(2)}$ with respect to their objective function values is possible. For some of these pairs, it can be observed that solution $x^{(1)}$ is at least equally good as $x^{(2)}$ with respect to all objective functions, and better than $x^{(2)}$ with respect to at least one objective function. In that case one solution *dominates* the other solution. If none of the two compared solutions dominates the other, the solutions are called *non-dominated* solutions. A solution $x^*$ is called *Pareto-optimal*, if there is no feasible solution that dominates $x^*$. The set of all image points $z^* \in Z$ of Pareto-optimal solutions $x^*$ is called the *Pareto-optimal/efficient frontier*. The following example illustrates both cases (we want to minimize $f_1$ and maximize $f_2$).

| Solution | $f_1$ | $f_2$ |
|---|---|---|
| A | 0 | 0 |
| B | 2 | 1.5 |
| C | 5 | 3 |
| D | 4 | 3.5 |
| E | 3 | 1.5 |

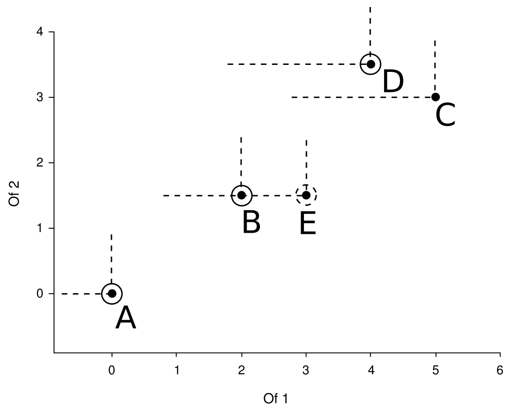



Figure 2.2.: Example solutions and corresponding image points in the objective space $Z$

When each two pairs of the four given solutions are compared it can be seen that solution $D$ dominates the solution $C$ and solution $B$ dominates solution $E$. The other solution $\{A, B, D\}$ form the set of non-dominated solutions. This curve is called the *Pareto-optimal front* all points on this curve are optimal solutions. Mathematical dominance and Pareto-optimality are defined as follows. To cover minimization and maximization problems the operator $\triangleright$ is used to compare solutions. $i \triangleright j$ denotes that solution $i$ is better than solution $j$ on a particular objective function. E.g. if the objective function should be minimized $i \triangleright j$ would mean "$i \leq j$".

*Definition* 2.1.1. A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$, if the following conditions are true:

1. The solution $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives, or
   $f_j\left(x^{(1)}\right) \ntriangleright f_j\left(x^{(2)}\right) \ \forall j = 1, 2, ..., t$.

2. The solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective, or
   $f_j\left(x^{(1)}\right) \triangleleft f_j\left(x^{(2)}\right)$ for at least one $j = 1, 2, ..., t$.

If one of the above defined conditions is not true, then the solution $x^{(1)}$ does not dominate solution $x^{(2)}$. Mathematically we write $x^{(1)} \preceq x^{(2)}$ if solution $x^{(1)}$ dominates solution $x^{(2)}$. Therefore either $x^{(1)}$ dominates $x^{(2)}$, or $x^{(1)}$ is non-dominated by $x^{(2)}$ or $x^{(1)}$ is non-inferior to $x^{(2)}$. In Cormen [27] the binary relation properties of the dominance operator are discussed. (i) Reflexive: The dominance relation is not reflexive, since any solution $x^{(1)}$ does not dominate itself. (ii) Symmetric: The dominance relation is not symmetric, because $x^{(1)} \preceq x^{(2)}$ does not imply $x^{(2)} \preceq x^{(1)}$. The opposite is true. If $x^{(1)}$ dominates $x^{(2)}$, then $x^{(2)}$ does not dominate $x^{(1)}$. The dominance relations is asymmetric. (iii) Antisymmetric: Since the dominance relation is not symmetric it cannot be antisymmetric. (iv) Transitive: The dominance relation is transitive. If $x^{(1)} \preceq x^{(2)}$ and $x^{(2)} \preceq x^{(3)}$, then $x^{(1)} \preceq x^{(3)}$.

In addition to the Definition 2.1.1 a second definition for dominance relationships exists. The relationship defined above is sometimes referred to as a *weak* dominance relation. A *strong* dominance relationship can be defined as follows.

*Definition* 2.1.2. A solution $x^{(1)}$ is said to strongly dominate the other solution $x^{(2)}$, if solution $x^{(1)}$ is strictly better than solution $x^{(2)}$ in all $t$ objectives.
$f_j\left(x^{(1)}\right) \triangleleft f_j\left(x^{(2)}\right) \ \forall j = 1, 2, ..., t$.

The definitions above can be used to define two different *non-dominated* sets. The *strongly* and the *weakly* non-dominated set.

*Definition* 2.1.3. Strongly non-dominated set: Among a set of solutions $P$, the strongly non-dominated set of solutions $P'$ are those that are not weakly dominated by any member of set $P$.

*Definition* 2.1.4. Weakly non-dominated set: Among a set of solutions $P$, the weakly non-dominated set of solutions $P'$ are those that are not strongly dominated by any member of set $P$.

Considering the example in Figure 2.2 it can be seen that solution $B$ weakly dominates solution $E$ and solution $D$ strongly dominates solution $C$. The strongly non-dominated is $\{A, B, D\}$,$\{A, B, D, E\}$ is the weakly non-dominated set. From examples and definitions, it can be seen that a weakly non-dominated set contains all solutions of the strongly non-dominated set, therefore the cardinality of the weakly non-dominated set is greater or equal to the cardinality of the strongly non-dominated set.

It is important that the feasible objective space not only contains non-dominated solutions. By using pair-wise comparison any finite set of solutions $P$ can be divided into the non-dominated set $P_1$ and the set of dominated solution $P_2$. For the non-dominated set $P_1$ the following conditions hold: (i) any two solutions of $P_1$ must be non-dominated with respect to each other, and (ii) any solution not belonging to $P_1$ is dominated by at least one solution in $P_1$. If the set $P$ is the entire feasible search space, the non-dominated set $P_1$ is called *Pareto-optimal* set.

*Definition* 2.1.5. The non-dominated set of the entire feasible search space $\tilde{S}$ is the globally Pareto-optimal set.

As in single-objective optimization, global and local optimal solutions can be identified. In multi-objective optimization, they are called global and local Pareto-optimal sets. For simplicity, we refer to the globally Pareto-optimal set as the Pareto-optimal set. A locally Pareto-optimal set can be defined as follows:

*Definition* 2.1.6. If for every member $x^{(1)}$ in a set $P$ there exists no solution $x^{(2)}$ (in the neighborhood of $x^{(1)}$ such that $||x^{(2)} - x^{(1)}||_\infty \leq \epsilon$, where $\epsilon \in \mathbb{R}^+$) dominating any member of the set $P$, then solutions of the set $P$ constitute a locally Pareto-optimal set.

### Objectives in Multi-Objective Optimization

The main goal in multi-objective optimization is to find a set of solutions that approximates the Pareto-optimal set as well as possible. As stated above, if conflicting objectives exist, usually the Pareto-optimal set contains more than one solution. If higher-level preference information is absent, all Pareto-optimal solutions are equally important. Therefore it is

important to find as many Pareto-optimal solutions as possible. Therefore the two goals of multi-objective optimization are: (i) To find a set of solutions as close as possible to the Pareto-optimal front, and (ii) to find a set of solutions as diverse as possible. The first goal corresponds to the goal in single-objective optimization. Different measures are available to measure the distance between the Pareto-optimal front and the solutions found.
The second goal is specific to multi-objective optimization. In addition to being converged close to the Pareto-optimal front, the solutions found must be sparsely spaced in the Pareto-optimal region. A diverse set of solutions assures a good set of trade-off solutions. "Diversity" can be defined in the decision space $X$ and the objective space $Z$, usually diversity in one space means diversity in the other space. If this is not the case, the task is to find a set of solutions having good diversity in the desired space. Also for this objective different measures are available.
An overview of possible measures for the quality of the approximation of the Pareto-optimal set is given in Section 2.3 Performance Assessment.

**Weighted Metric Method** The weighted metric method scalarizes a set of objective by using weighted Minkowski distances of an of any solution $x$ to the ideal point $z^*$. One problem of this approach is the definition of the weights of the objectives. These distances can be minimized as follows:

$$\begin{aligned}
\min l_p(x) &= \left(\sum_{k=1}^{t} w_k |f_k(x) - z_k^*|^p\right)^{1/p} && k = 1, 2, \ldots, t, && (2.2)\\
\text{s.t. } g_j(x) &\geq 0 && j = 1, 2, \ldots, \bar{m}, \\
h_j(x) &= 0 && j = \bar{m} + 1, \ldots, m, \\
x_i^L \leq x_i &\leq x_i^U && i = 1, 2, \ldots, n,
\end{aligned}$$

where $w_k$ (and in most cases $\in [0, 1]$) is the weight for the k-th objective. It is the usual practice that the sum of weights equals one. The parameter $p$ can take a value between 1 and $\infty$. This parameter influences the measure that is used. The following list provides an overview of commonly used values for $p$ and the resulting optimization problems.

- $p = 1$: the problem is equivalent to the standard weighted sum approach
- $p = 2$: a weighted Euclidean distance is minimized
- $p = \infty$: the *weighted Chebyshev* problem, the largest deviation $|f_k(x) - z_k^*|$ should be minimized.

Let us discuss two the two cases (i) $p = 1$ and (ii) $p = \infty$.

(i) Weighted sum approach $p = 1$: two interesting properties of the problem shown in (2.2) and reproduced from Miettinen[90] are:

**Theorem 2.1.7.** *The solution to the problem represented by* (2.2) *is Pareto-optimal if the weight is positive for all objectives.*

This is true for each MOOP but does not imply that any Pareto-optimal solution can be found by using a positive weight vector. This is only true for convex problems. For the proof of the following problem refer to Miettinen[90].

**Theorem 2.1.8.** *If $x^*$ is a Pareto-optimal solution of a convex multi-objective optimization problem, then there exists a non-zero positive weight vector $w$ such that $x^*$ is a solution to the problem given by equation* (2.2).

The following two figures represent two cases of minimization problems with two objective functions. Figure 2.3 represents a convex optimization problem, Figure 2.4 shows a non-convex optimization problem.

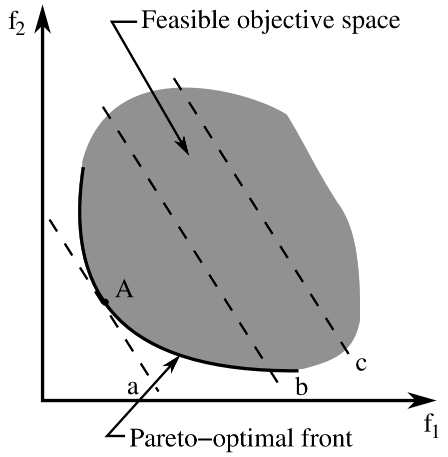



Figure 2.3.: Weighted sum approach on a convex Pareto-optimal front

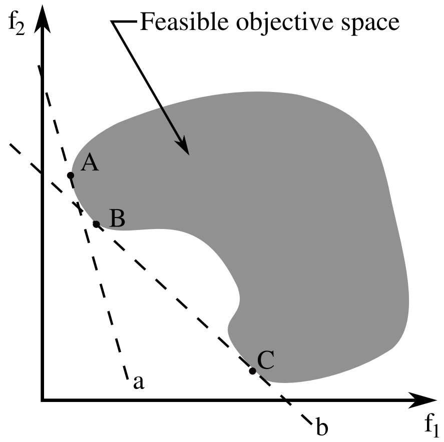



Figure 2.4.: Weighted sum approach on a non-convex Pareto-optimal front

For convex optimization problems all points on the Pareto-optimal front can be found by using a weighted sum approach. The contour lines ‘a’,‘b’,‘c’ represent the objective function $F$. As $F$ is a linear combination of the objectives $f_1$ and $f_2$, every point on the contour line will have the same value for $F$. The slope of the contour lines is related to the weights $w_1, w_2$, the location depends on the value for $F$. The minimum value can be found at point ‘A’. Different weight vectors will yield different optimal solutions, these would always lie on the Pareto-optimal front.

For non-convex MOOP the weighted sum approach cannot find certain Pareto-optimal solutions. In Figure 2.4 the points ‘A’, ‘B’, and ‘C’ found by using weights that yield the contour lines ‘a’ and ‘b’, will correspond to Pareto-optimal solutions. But there will not be any weight vector that will create a contour line that is a tangent to any point between ‘B’ and ‘C’, such that this contour line will not be a tangent to another better point (point with smaller value for $F$) in the objective space. Therefore the weighted sum approach only can find solutions that lie on the *convex hull* of the feasible objective space. These solutions are called *supported*. But in general not all Pareto-optimal solutions are supported. To overcome the problem of the weighted sum approach the weighted Chebyshev distance can be used.

(ii) $p = \infty$ Weighted Chebyshev approach: The following figures illustrate the differences of weighted metric methods with $p = 1$ and $p = \infty$, optimal solutions for two different weight vectors are shown.

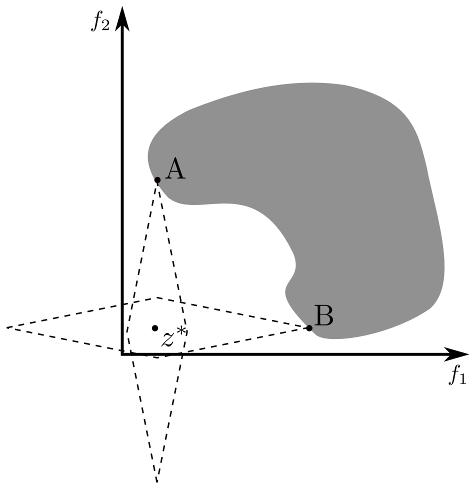



Figure 2.5.: Weighted metric method $p = 1$

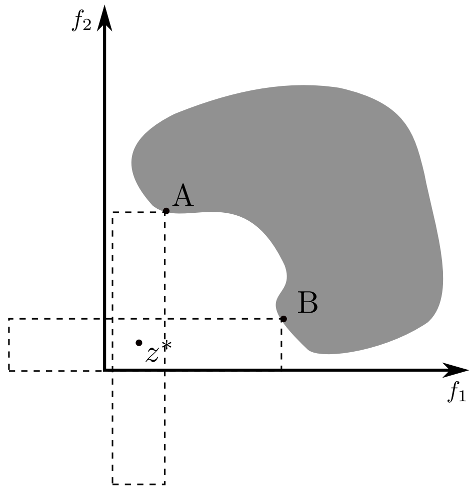



Figure 2.6.: Weighted metric method $p = \infty$

Figure 2.5 shows that, as already mentioned no solution in the region $BC$ can be found using a weighted sum approach. However, as illustrated in Figure 2.6, any Pareto-optimal solution can be found by using the weighted Chebyshev metric. One problem that arises if large $p$ values are used is that the problem becomes nondifferentiable, therefore most times gradient-based methods can not be used to find the minimum solution.

By using weighted metric methods multi-objective optimization problems can be transfered into different (depending on the chosen weights) single-objective optimization problems. A multi-objective optimizer can therefore obtain either the Pareto-optimal set or at least an approximation of it, by solving a sequence of single-objective optimization problems with varying weight vectors, e.g. Ralphs et al.[104] present an iterative method that determines the Pareto-optimal set of bi-objective mixed integer linear optimization problems by using weighted Chebyshev distances.

## 2.2. Simulation

Let a stochastic optimization problem

$$\max \{f(x) \,|x \in S\} \tag{2.3}$$

with $f(x) = \mathbb{E}(f(x,\omega))$ be given, where $S$ is a finite decision space, and $\omega$ denotes the influence of randomness. Already in this simplest form of a stochastic problem the evaluation of the function $f(x)$ can be quite complicated. The reason is that in general the problem is one of numerical integration in high dimensions corresponding to the random variables $\omega$ (cf. (Birge and Louveaux[19])). This general problem requires some form of approximation. One possible approach is the use of Monte Carlo (MC) simulation, which has favorable properties for higher dimensions. By using MC, confidence intervals on the results can be obtained. For getting an estimate of $\mathbb{E}(f(x,\omega))$, we use sampling by drawing $N$ random *scenarios* $\omega_1, \ldots, \omega_N$ independently from each other, each scenario $\omega_\nu$ consists of a vector of $U^{(\nu)} = \left(U_1^{(\nu)}, \ldots, U_m^{(\nu)}\right)$ of i.i.d. random numbers $U_i^{(\nu)}$ distributed according to a density $h_i(\cdot)$ $(i = 1, \ldots, m)$ that can take values in the space $A$. Then, the *sample average estimate* of $f(x) = \mathbb{E}(f(x,\omega))$ is given by

$$\hat{f}(x) = \frac{1}{N} \sum_{\nu=1}^{N} f(x, \omega_\nu) \approx \mathbb{E}(f(x,\omega)).$$

Evidently, the sample average estimate is an unbiased estimator for $f(x)$. Denoting the standard deviation of $f(x,\omega)$ by $\sigma(x)$, the standard deviation of $\hat{f}(x)$ ($\sigma_{\hat{f}(x)} = \frac{\sigma(x)}{\sqrt{N}}$) can be approximated by using the sample variance

$$\hat{s}^2(x) = \frac{1}{N-1} \sum_{\nu=1}^{N} \left(f(x,\omega_\nu) - \hat{f}(x)\right)^2 \approx \sigma^2(x).$$

An approximation for the standard deviation of the estimator $\hat{f}(x)$ is given by

$$\hat{s}_{\hat{f}(x)} = \frac{\hat{s}(x)}{\sqrt{N}}.$$

Which can be used to define confidence intervals for the estimator $\hat{f}(x)$ of $f(x)$. A possible solution to reduce the variance of $\hat{f}(x)$ would be to increase the sample size $N$, but computational efforts would increase quadratically (e.g. to decrease the standard deviation by a factor 10, sample sizes would have to increase by a factor 100). To reduce the the variance of $\hat{f}(x)$ without paying the costs of increased sample sizes, *importance sampling* (see, e.g.[107]) (IS) can be used. The basic idea of importance sampling consists in changing the distribution $h_i(\cdot)$ of the random variable(s) on which the sampling is based to a distribution $h_i{}^*(\cdot)$, such that the more interesting events (events that have a large influence on the estimator) can be observed more frequently. To compensate for the change of the distributions, weighing each output by the so-called likelihood ratio, which is given as the quotient of true probability (or density) and changed probability(or density), is used to ensure that the procedure preserves the unbiasedness of the estimator.

$$\mathbb{E}(f(x,\omega)) = \int_A f(x,\omega)\, h(\omega)\, d\omega = \int_A f(x,\omega)\, L(\omega)\, h^*(\omega)\, d\omega,$$

where $L(\omega) = \frac{h(\omega)}{h^*(\omega)}$ denotes the likelihood ratio.

In general (cf. Fishman[42]) $h^*(\omega)$ should be chosen as proportional to $f(x,\omega)\, h(\omega)$ as possible for each $\omega \in A$.

## 2.3. Performance Assessment

In this section, methods for the comparison of the performance of stochastic multi-objective optimizers (MOO) over several runs, as well as methods used to obtain quantitative and statistically sound inferences are presented (based on Knowles et al.[75]). Performance of a MOO involves the quality of the solutions found and the time needed to generate these solutions. In the case of stochastic optimizers the relation between time and quality is not fixed but can be described by a probability density function. Therefore every statement about the performance is probabilistic. Furthermore the outcomes of a MOO are usually not single values but a set of trade-offs. This two questions (i) how to define the quality of the solutions and (ii) how to represent the outcomes of multiple runs in terms of probability density functions. As stated before each MOO returns a set of mutually incomparable solutions. It is not guaranteed that this solution set is the true Pareto-optimal set of the optimization problem, therefore this set is called a *Pareto front approximation* or *approximation set*.

In section 2.1.1 Pareto-optimal solutions, weak and strong Pareto dominance and some preference relations are defined. A finer grained listing of preference relationships between two points in the objective space is shown in Table 2.1 below.

| relation | | interpretation in objective space |
|---|---|---|
| strictly dominates | $z^{(1)} \prec\prec z^{(2)}$ | $z^{(1)}$ is better than $z^{(2)}$ in all objectives |
| dominates | $z^{(1)} \prec z^{(2)}$ | $z^{(1)}$ is not worse than $z^{(2)}$ in all objectives and better in at least one objective |
| weakly dominates | $z^{(1)} \preceq z^{(2)}$ | $z^{(1)}$ is not worse than $z^{(2)}$ in all objectives |
| incomparable | $z^{(1)} \| z^{(2)}$ | neither $z^{(1)} \preceq z^{(2)}$ nor $z^{(2)} \preceq z^{(1)}$ |
| indifferent | $z^{(1)} \sim z^{(2)}$ | $z^{(1)}$ has the same value as $z^{(2)}$ in all objectives |

Table 2.1.: Preference relations (Knowles et al.[75])

The preference relations of solutions defined in Table 2.1 above can be extended to Pareto set approximations. In the following Table 2.2 the selected relations are shown.

| relation | | interpretation in objective space |
|---|---|---|
| strictly dominates | $A \prec\prec A$ | every $z^{(2)} \in B$ is strictly dominated by at least one $z^{(1)} \in A$ |
| dominates | $A \prec B$ | every $z^{(2)} \in B$ is dominated by at least one $z^{(1)} \in A$ |
| better | $A \lhd B$ | every $z^{(2)} \in B$ is weakly dominated by at least one $z^{(1)} \in A$ and $A \nsim B$ |
| weakly dominates | $A \preceq B$ | every $z^{(2)} \in B$ is weakly dominated by at least one $z^{(1)} \in A$ |
| incomparable | $A \| B$ | neither $A \preceq B$ nor $B \preceq A$ |
| indifferent | $A \sim B$ | $A \preceq B$ and $B \preceq A$ |

Table 2.2.: Selected preference relations on Pareto front approximations, where $A$ and $B$ are two different Pareto front approximations (Knowles et al.[75])

Quality assessment in the context of multi-objective optimizers is usually done in objective space; the aim of an MOO is to find a Pareto set approximation as "close" as possible to the "true" Pareto front of the optimization problem and to, a certain extent the spread of the solutions across the objective space. The quality assessment process has two stages (i) sample transformation: the samples of approximation sets are first transfered into another representation (e.g. a sample of indicator values (*quality indicators*), an empirical attainment function, or a sample of ranks), and (ii) statistical testing is used to answer the question, whether the approximation set distribution of optimizer A is better than the approximation set distribution of optimizer B.

### 2.3.1. Sample Transformations

**Quality Indicators:** In general a unary *quality indicator* $I$ is defined as a mapping of all approximation sets $\Omega$ to the set of real numbers. $I$ establishes an order on $\Omega$ that represents the quality of approximation sets. The difference between $I(A)$ and $I(B)$ reveals a difference in the quality of the two sets. This information goes beyond Pareto dominance; additional knowledge (*preference information*) can be represented. It is possible that if two different indicators $I_1$ and $I_2$ are used the following result may arise: $I_1(A)$ better than $I_1(B)$ and $I_2(B)$ better than $I_2(A)$. This shows that all comparisons of MOO's are not restricted to benchmark problems and parameter settings only but also to the considered quality indicators. An important property of quality indicators is whether or not they are *Pareto compliant*. If an indicator is Pareto compliant, then whenever a set $A$ is preferred to a set $B$ with respect to weak Pareto dominance, then $I(A)$ is at least as good as $I(B)$. More information can be found in Knowles et al.[75]. In addition to the unary quality indicators, indicators that can take an arbitrary number of approximation set as arguments can be designed. In this work we also use binary quality indicators that assign real numbers to pairs of approximation set.
(1) The *hypervolume* $I_H$ measure (see Zitzler and Thiele[127], Zitzler et al.[128,129]) which is defined as the volume of the objective space dominated by an approximation set. For the calculation of $I_H$, the objective space must be bounded; if this is not the case then a bounding reference point must be used. In addition to the (absolute) hypervolume, one can also compute the *relative* hypervolume $I_H^-$ which is defined as the difference between the (absolute) hypervolume of the reference set and that of the approximation set. The relative hypervolume is small in the case of a good approximation set. When a normalization (described below) of the objectives is used, then the higher the hypervolume, the better the approximation set. The hypervolume indicator has some desirable theoretical properties: Whenever $A \triangleleft B$, then $I_H(A) > I_H(B)$ therefore from $I_H(A) < I_H(B)$, one can infer that $A$ cannot be better than $B$.
(2) The *epsilon indicators* $I_{\epsilon+}^1$ resp. $I_{\epsilon}^1$ (see Zitzler and Thiele[127], Zitzler et al.[128,129]) give the minimum term $\epsilon$ resp. the minimum factor $\epsilon$ by which each point of an approximation set $B$ in the objective space can be shifted by component-wise addition resp. by component-wise multiplication, such that the resulting set is weakly dominated by a reference set $A$. The smaller an epsilon indicator, the better is the approximation set. The

set of Pareto-optimal solutions has $I_{\epsilon^+} = 0$ and $I_\epsilon = 1$.

$$I_\epsilon(A,B) = \inf_{\epsilon\in\mathbb{R}} \left\{ \forall z^{(2)} \in B \; \exists z^{(1)} \in A : z^{(1)} \preceq_\epsilon z^{(2)} \right\},$$

where in the case of a minimization problem the multiplicative $\epsilon$-dominance relation is defined as:

$$z^{(1)} \preceq_\epsilon z^{(2)} \Leftrightarrow \forall i \in 1,\ldots,n : z_i^{(1)} \leq \epsilon \cdot z_i^{(2)}$$

The additive epsilon indicator $I_{\epsilon^+}$, is defined using the additive $\epsilon$-dominance.

$$z^{(1)} \preceq_\epsilon z^{(2)} \Leftrightarrow \forall i \in 1,\ldots,n : z_i^{(1)} \leq \epsilon + z_i^{(2)}$$

The unary versions $I_\epsilon^1$ and $I_{\epsilon^+}^1$ can be defined by using a reference set $R$:

$$I_\epsilon^1 = I_\epsilon(A,R)$$

Both indicators should be minimized. $I_\epsilon^1 < 1$ or $I_{\epsilon^+}^1 < 0$ implies that $A$ strictly dominates the reference set $R$. Whenever $A \triangleleft B$, than $I_\epsilon^1(A) \leq I_\epsilon^1(B)$ respectively $I_{\epsilon^+}^1(A) \leq I_{\epsilon^+}^1(B)$. If the hypervolume and the $\epsilon$-indicators return opposite preference orderings, then the sets are incomparable.
(3) The *spacing* $I_{SP}$ measure (measure $Q_5$ in Jaszkiewicz[69]) which quantifies the quality of the distribution of the proposed efficient solutions in objective space and is the lower, the better this distribution is.

$$I_{SP}(A) = \sqrt{\frac{1}{|A|-1}\sum_{z\in A}\left(\bar{d} - d(z)\right)^2}, \text{where } d(z) = \min_{\acute{z}\in A}\left\{\sum_{i=1}^{n} |z_i - \acute{z}_i|\right\}$$

(4) The *coverage* $I_{CO}$ measure (measure $Q_6$ in Jaszkiewicz[69]). It is a binary indicator; $I_{CO}(A,B)$ gives the share of solutions in set $B$ that are dominated by the solutions in set $A$.

$$I_{CO}(A,B) = \frac{|\left\{z^{(2)} \in B\right\}|\exists z^{(1)} : z^{(1)} \preceq z^{(2)}|}{|B|}$$

Considering computational experiments where multiple ($k$) runs were performed for each test case and each algorithm, we computed the average $I_{CO}(1)$ of all $k^2$ coverage values $I_{CO}(A_i, B_j)$ for $i,j = 1,\ldots,k$ and the average $I_{CO}(2)$ of all coverage values $I_{CO}(B_i, A_j)$

for $i, j = 1, \ldots, k$, where $A_i$ and $B_i$ denotes the solution set provided by the $i$th run of the first and of the second considered algorithm, respectively. If $I_{CO}(1) > I_{CO}(2)$, the first algorithm is better than the second, and vice versa.
An advantage of quality indicators is that comparative studies are easy to accomplish because the samples of approximations are transformed into samples of real values for which standard statistical testing procedures exist. Quantitative statements are possible. A drawback of this approach is the loss of generality because every indicator represents a specific preference information.

**Empirical Attainment Function:** *Attainment functions* take into account that meta-heuristics for multi-objective problems are usually *stochastic* algorithms, i.e., in general, they produce different solutions at each run. These results ca be described by the distribution of an outcome set $A = \left\{a^{(i)} \in \mathbb{R}^n, i = 1, 2, \ldots, M\right\}$ with cardinality $M$. In order to capture this aspect in formal terms, one defines for each point $z$ (called *goal*) in the objective space $Z = \mathbb{R}^n$ the probability $\alpha_A(z)$ that the optimizer attains goal $z$ in a single run. A goal (objective vector) is attained if it is weakly dominated by the approximation set. In formal terms $z$ is attained by approximation set $A$ if $a \preceq z$ for at least one $a \in A$. Using the notation $A \preceq \{z\}$ iff there exists an $a \in A$ such that $a \preceq z$, one can write $\alpha_A(z) = P(A \preceq \{z\})$. The function $\alpha_A(z)$ is called the *attainment function*. An estimate for the attainment function is obtained by performing $N$ runs of the algorithm and setting $\alpha_N(z) = (1/N) \sum_{\nu=1}^{N} I(A_\nu \preceq \{z\})$, where $A_\nu$ is the approximation set of run $\nu$ of the optimizer, and $I(.)$ is an indicator function that evaluates to one if the argument of the function is true and else to zero. The function $\alpha_N(z)$ is called the *empirical attainment function* (EAF). The EAF gives the relative frequency for each objective vector that was attained. By using the EAF it is possible to visualize i.e. all goals that have been attained in at least 50 % of the runs. The *k % approximation set* of the EAF is defined as the set of all goals $z$ that are attained in at least $k$ % of the $N$ runs. Formaly a k%-approximation $A$ set is defined as:

$$A = \{\forall z \in Z : \alpha_N(z) \geq k/100\} \tag{2.4}$$

The *attainment surface* defined as the union of the tightest goals that are known to be attainable by an approximation set $A$, formally $\{z \in Z : A \preceq z \wedge \neg A \prec\prec z\}$. The $p\%$ attainment surface represents the border of the area of all those points in the plane for which $\alpha(z)$ is at least $p\%$. Thus, the 100% attainment surface gives the border of the points dominated by the proposed solution sets of *each* run, etc. These surfaces can be used for visualizing the the outcomes of multiple runs of the optimizer. Attainment functions therefore allow to visually judge whether a metaheuristic algorithm is stable with respect to the influence of the different streams of random numbers in different runs. The advantage of using the attainment function approach is, that less information is lost than when quality indicators or dominance ranking are used. This is achieved by the fact that the transformed samples are multidimensional. Visualization allows a deeper insight into the strengths and weaknesses of an optimizer. A major drawback is that this approach is computationally expensive. More information on the attainment function can be found in the papers by Fonseca et al. [43], Grunert da Fonseca et al. [50], Zitzler and Thiele [127], Zitzler et al. [128,129].

**Reference Points and Sets** For some quality measures, a reference point $z^+$ or a reference set is needed. For example for the hypervolume metric the dominated region has to be bounded by a reference point. All generated approximations $A_1, A_2, \dots, A_r$ have to be dimension-wise worse than the reference point:

$$\forall i = 1 \dots r \; \forall z \in A_j \; \forall j = 1 \dots n : z_j \triangleright z_j^+$$

For some approaches a reference set is needed, e.g. epsilon indicators. The best reference set would be the "true" Pareto front. However in most cases the Pareto front cannot be computed in reasonable time. In this case two methods are recommended by Knowles et al. [75]. (i) The reference set is the non-dominated set of the union of all approximation sets. (ii) The reference set is the set that dominates 50% of solutions in the search space. This can be generated by, for example creating 1000 points randomly (each representing one outcome of a random search strategy) and then taking the 50% attainment surface as the reference set.

**Normalization** Pareto dominance in general is independent of scales and normalization. For some of the indicators (e.g. $I_H$, and $I^1_{\epsilon+}$) normalization is necessary to allow equal contribution of the different objectives. To obtain comparable magnitudes of values for the objective functions, we rescaled both objective function values to the interval $[1, 2]$[1]. The following equation is used:

$$z_i' = \frac{z_i - z_i^{(min)}}{z_i^{(max)} - z_i^{(min)}} + 1, \tag{2.5}$$

[1]The interval $[1, 2]$ is used instead of $[0, 1]$ in order to facilitate the calculation of $I^1_\epsilon$ (to avoid divisions by zero)

where $z_i^{(min)}$ and $z_i^{(min)}$ are the corresponding minimum and maximum values of objective $i$. These can be obtained from the reference set.

### 2.3.2. Statistical Testing

To describe and determine if an approximation set distribution of an optimizer is better than another, statistical methods are used. *Descriptive statistics* are convenient to summarize random samples from distributions. *First order moments*, e.g. mean, median and mode, describe the *location* of the distribution on the real number line. *Second-order moments*, e.g. variance, standard deviation and inter-quartile range, describe the spread of the data. Box-plots or tables of mean and standard deviation provide a good overview. Descriptive statistics do not provide enough information to decide if the approximation set distribution of one optimizer is better than another. *Statistical inference* methods must be used.

The fact that only a limited number of samples is available, prohibits definitive judgments. *Statistical tests* test how likely a certain *null hypotheses* ($H_0$) is true. An example for a $H_0$ is: "samples A and B are drawn from the same distribution". The result of a statistical test is called *p-value*. The *significance level* (often $\alpha$) defines the largest acceptable p-value. This threshold is user defined. If the p-value is lower than the significance level, then this indicates that the $H_0$ can be rejected. An *alternative hypothesis* $H_A$ can be chosen at a significance level $\alpha$. In tests where it is assumed that the samples are drawn from a distribution that closely approximates a known distribution e.g. normal distribution defined by its parameters, are called *parametric statistical tests*. These tests although potentially powerful, mostly can not be used for stochastic optimizer outputs because the samples are usually too small. To solve this problem *nonparametric tests* e.g. rank tests and permutation tests can be used.

To test single quality indicators, standard univariate statistical tests can be used, for the comparison of two optimizers: e.g. Man-Whitney rank sum test, Fischer's permutation test and for more than two optimizers the Kruskal-Wallis test. As the EAF is a generalization of a empirical univariate cumulative distribution function (ECDF). The Kolmogorov-Smirnov (KS) test can be used to determine if two ECDF's are different. This test only reveals if there is a difference between the EAF's; in order to probe specific difference between EAF's visualization methods should be used. More information can be found in Knowles et al.[75].

### 2.3.3. Running Performance Metrics

The aforementioned metrics can be used to compare the results of two or more MOO purely on the basis of the results that have been obtained at the end of a simulation run. Interesting information on the internal behavior of a MOO (how the MOO generates the final result), is not captured by these measures. In most single-objective studies (using heuristics) analyze the internal behavior of the optimizer by using measures that show how the solution quality evolves with time. Deb and Jain [31] demonstrate the use of two running performance metrics to investigate the internal behavior of multi-objective optimizers. This information provides an insight to the working of the optimizer and facilitates to decide if a problem is easy or difficult for the considered optimizer. Considering the goals in multi-objective optimization two metrics are interesting: (i) one for measuring the convergence of the solutions of the current non-dominated set $P^{(t)}$ to the Pareto-optimal front $P^*$ (or a reference set $R$) and (ii) a measure to describe the diversity of solutions.
(i) Convergence (Distance), can be easily assessed by the average normalized Euclidean distances of each point of the current non-dominated set $P^{(t)}$ to the closest point in the reference set $R$.

$$I_C\left(P^{(t)}\right) = \frac{\sum_{z \in P^{(t)}} d\left(z\right)}{|P^{(t)}|}, \text{where } d\left(z\right) = \min_{\acute{z} \in R} \sqrt{\sum_{i=1}^{n} \left( \frac{z_i - \acute{z}_i}{z_i^{(max)} - z_i^{(min)}} \right)^2}$$

$z_i^{(min)}$ and $z_i^{(min)}$ are the corresponding minimum and maximum values of objective $i$ in the reference set $R$. Usually $I_C\left(P^{(t)}\right)$ is normalized to keep the metric within $[0,1]$ by dividing by its maximum value $\bar{I_C}\left(P^{(t)}\right) = I_C\left(P^{(t)}\right) / \max_{t=0}^{T} I_C\left(P^{(t)}\right)$.
(ii) Diversity is assessed by projecting the current non-dominated points on a suitable hyper-plane. Each hyper-plane is divided into several small grids. Depending on which grids contain a non-dominated point, a diversity metric is defined. The best possible diversity measure value is achieved if each intervals is represented by at least one point. If this is not the case, a moving window is used to define the balance of the distribution of empty and non-empty intervals. A detailed description of this measure is given in Deb and Jain [31].

## 2.4. Algorithms

### 2.4.1. Adaptive Pareto-Sampling Algorithm

In the following, we shortly recapitulate the Adaptive Pareto-Sampling (APS) approach by Gutjahr [54] for multi-objective stochastic combinatorial optimization problems. Let a multi-objective stochastic combinatorial optimization problem

$$\max \left(f^{(1)}(x), \ldots, f^{(p)}(x)\right) \qquad (2.6)$$
$$\text{s.t. } x \in S$$

with $f^{(\vartheta)}(x) = \mathbb{E}(f^{(\vartheta)}(x,\omega))$ $(\vartheta = 1, \ldots, p)$ be given, where $S$ is a finite decision space, and $\omega$ denotes the influence of randomness. For getting an estimate of $\mathbb{E}(f^{(\vartheta)}(x,\omega))$, we use sampling by drawing $N$ random *scenarios* $\omega_1, \ldots, \omega_N$ independently from each other. Then, the *sample average estimate* of $f^{(\vartheta)}(x) = \mathbb{E}(f^{(\vartheta)}(x,\omega))$ is given by

$$\frac{1}{N}\sum_{\nu=1}^{N} f^{(\vartheta)}(x,\omega_\nu) \approx \mathbb{E}(f^{(\vartheta)}(x,\omega)). \qquad (2.7)$$

As stated in Section 2.2, the sample average estimate is an unbiased estimator for $f^{(\vartheta)}(x)$. An approximation to the solution of the given problem (2.6) can be computed by solving a related problem where the expectations forming the objective functions are replaced by sample average estimates with some sample size $N$. In this way, we obtain the following deterministic multi-objective problem:

$$\max \left(\frac{1}{N}\sum_{\nu=1}^{N} f^{(1)}(x,\omega_\nu), \ldots, \frac{1}{N}\sum_{\nu=1}^{N} f^{(p)}(x,\omega_\nu)\right) \qquad (2.8)$$
$$\text{s.t. } x \in S$$

We call problem (2.8) the *bicriteria sample average approximation* (BSAA) problem corresponding to the original problem (2.6).

In Algorithm 2.4.1, we present the pseudocode of the APS algorithm. The algorithm is iterative and works with a current solution set $L^{(\kappa)}$ which is updated from iteration to iteration. In each of these iterations, first of all a corresponding deterministic BSAA problem is solved in order to obtain a *proposal* for the solution set. After that, the elements of the solution of the BSAA problem are merged with those contained in $L^{(\kappa-1)}$, the elements in the union of both sets are *evaluated* based on independent samples for each solution and each objective function, and dominated elements (w.r.t. the evaluation results) are eliminated. This yields the new solution set. The sample sizes are controlled by sequences $(s_\kappa)$ and $(\bar{s}_\kappa)$ of positive integers ($\kappa = 1, 2, \ldots$).

**Algorithm 2.4.1:** APS

initialize the solution set $L^{(0)}$ as the empty set;
**for** *iteration* $\kappa = 1, 2, \ldots$ **do**
  (a) "solution proposal":
    draw a sample $\{\omega_1, \ldots, \omega_{s_\kappa}\}$ of $s_\kappa$ scenarios;
    for the drawn sample, determine the Pareto-optimal set $S^{(\kappa)}$ of the BSAA problem with sample size $s_\kappa$;

  (b) "solution evaluation":
    **foreach** $x \in L^{(\kappa-1)} \cup S^{(\kappa)}$ *and each* $\vartheta = 1, \ldots, p$ **do**
      draw an independent sample $\{\omega_1(x,\vartheta), \ldots, \omega_{\bar{s}_\kappa}(x,\vartheta)\}$ of $\bar{s}_\kappa$;
      based on this sample, determine an estimate of $f^{(\vartheta)}(x)$ ;
    **end**
    obtain $L^{(\kappa)}$ as the set of Pareto-optimal solutions in $L^{(\kappa-1)} \cup S^{(\kappa)}$ according to the objective function estimates just determined;

**end**
**Output**: set of Pareto-optimal decision vectors $L^{(\kappa)}$

The determination of the Pareto-optimal set $S^{(\kappa)}$ in part (a) of APS can either be performed by an exact algorithm (e.g., if the deterministic problem has the structure of a bi-objective integer linear program, the algorithm by Chalmet et al. [24] or (adaptive) $\varepsilon$-constraint method by Haimes et al. [59], Laumanns et al. [81] can be applied for this purpose), or alternatively by a (multi-objective) metaheuristic. For a general discussion of convergence properties of the proposed method see Gutjahr [54].

### 2.4.2. Nondominated Sorting Genetic Algorithm II

Nondominated Sorting Genetic Algorithm II (NSGA-II) by Deb et al. [33] is a genetic algorithm searching for an approximation to the Pareto set of a multi-objective optimization problem by the successive computation of a series of generations of solutions. In each iteration, the algorithm computes a new generation from the current one by using three algorithmic components: (i) fast-non-dominated-sort, which is an efficient algorithm for partitioning a set of solutions into so-called non-dominated fronts (the first non-dominated is the set of non-dominated solutions, the second non-dominated front is the set of non-dominated solutions after removal of the first non-dominated front, etc.), (ii) a rank assignment method assessing the quality of a solution with respect to the aim that points on the approximated Pareto front should be well-distributed, and (iii) the genetic operators crossover, mutation, and selection.

In Algorithm 2.4.2, we present the pseudocode of the NSGA-II algorithm. Basically, NSGA-II works as follows: In the initialization phase, a population $P_0$ of $M_0$ solutions is generated. All solutions are evaluated with respect to the objective functions. Then, fast-non-dominated sort is applied to $P_0$, and the genetic operators crossover and mutation are used to derive an offspring population $Q_0$ of size $M_0$. In each iteration $\mu = 0, 1, \ldots$, the following operations are applied until a termination criterion is fulfilled. The two sub-populations $P_\mu$ and $Q_\mu$ are joined to a population $R_\mu$ of size $2M_0$. The new parent population $P_{\mu+1}$ is created by first performing non-dominated-sort to $R_\mu$ and then successively copying the solutions to $P_\mu + 1$ in the order given by the obtained non-dominated fronts, until $P_\mu + 1$ is full, i.e., contains $M_0$ elements. For the last non-dominated front that can be taken into account before $P_\mu + 1$ is full, the choice of the solutions is based on the rank indices obtained from the rank assignment method. Finally, crossover and mutation are applied to the current parent population $P_\mu + 1$ in order to generate the offspring population $Q_{\mu+1}$.

**Algorithm 2.4.2:** NSGA-II

$P_0$ = create-initial-parent-pop ();
$F$ = fast-nondominated-sort ($P_0$);
$Q_0$ = make-new-pop ($F$);
$\mu = 0$;
**repeat**
  $R_\mu = P_\mu \cup Q_\mu$;
  $F$ = fast-nondominated-sort ($R_\mu$);
  $P_{\mu+1}$ = select-new-parents ($F$);
  $Q_{\mu+1}$ = make-new-pop ($P_{\mu+1}$);
  $\mu = \mu + 1$;
**until** *termination*;
$R_\mu = P_\mu \cup Q_\mu$;
$F$ = fast-nondominated-sort ($R_\mu$);
**Output**: set of Pareto-optimal decision vectors $F_0$

**Constraint handling.** In general, constraints divide the search space into feasible and infeasible regions, therefore all Pareto-optimal solutions must be feasible. Different ways exists to handle constraints, e.g. ignoring infeasible solutions by excluding them from the search process, repair mechanism, penalty function approaches and the constrained tournament method (cf. Deb [29]). In this work we apply two different approaches to handle constraints. In the first application we use a repair mechanism. In the second application the constrained tournament method (Deb et al. [32]) is used. Considering a given solution $x$ and constraints of the form $g_j(x) \geq 0$ $(j = 1, \dots, J)$, the constraint violation $\omega_j(x)$ is defined as $\omega_j(x) = |g_j(x)|$ if $g_j(x) < 0$, and as $\omega_j(x) = 0$ otherwise. To calculate the overall constraint violation $\Omega(x)$ of a solution $x$, the constraint violations for the (normalized) constraints are added: $\Omega(x) = \sum_{j=1}^{J} \omega_j(x)$.

The relation of *constrain-domination* is defined in Deb et al. [32] as follows. For two solutions $x^i$ and $x^j$, solution $x^i$ is said to *constrain-dominate* solution $x^j$, if one of the following conditions is satisfied: (i) Solution $x^i$ is feasible and solution $x^j$ is not. (ii) Both solutions $x^i, x^j$ are infeasible, and $\Omega(x^i) < \Omega(x^j)$. (iii) Both solutions $x^i, x^j$ are feasible, and solutions $x^i$ dominates solution solutions $x^j$ in the usual sense. During the non-dominated sorting procedure of NSGA-II, a solution $x^i$ that constrain-dominates a solution $x^j$ is preferred to $x^j$.

### 2.4.3. Pareto Ant Colony Optimization

The Pareto Ant Colony Optimization (P-ACO) algorithm is a multi-objective metaheuristic introduced in Doerner et al. [34,35]. The P-ACO algorithm generalizes the Ant Colony Optimization (ACO) metaheuristic (see Dorigo and Stützle [36]) for single-objective problems to the case of several objective functions, determining approximations to the set of Pareto-efficient solutions. The special variant of P-ACO (adopted from Gutjahr [51]) used in this work is slightly different from that in the original papers Doerner et al. [34,35]. ACO is a nature-inspired metaheuristic where solutions are constructed randomly and step-by-step. To encode a solution all ACO algorithms use a *construction graph* $\mathcal{C}$. In general construction steps that have turned out as part of good solutions in previous iterations of the construction process are favored via "pheromone values" during the current iteration. As stated before ACO constructs solutions $x$ iteratively. Each solution is represented by a feasible walk through the construction graph. The construction process stops if there is no feasible unvisited successor node available. The computational agent that constructs a solution is called an ant.

The decision as to which feasible successor node $l$ of a node $k$ should be included in the walk, depends on the *pheromone value* $\tau_{kl}$, and the *visibility* $\eta_{kl}$. The pheromone value is a memory that stores the suitability of step $(k, l)$ in previous runs; the visibility represents a pre-evaluation based on a problem specific heuristic. The visibility may depend

on the partial walk $u$ that the ant has performed so far. The probability $p_{kl}$ that an ant chooses the edge $(k, l)$ is proportional to $[\tau_{kl}]^{\alpha} [\eta_{kl}(u)]^{\beta}$ , where $\alpha$ and $\beta$ are parameters, determining the relative influence of the pheromone trail and the heuristic information. Generally in each iteration of an ACO algorithm a certain number of random walks of ants are performed; these walks form a *round*. Depending on the implementation, either the round winner, the global best solution or a number of ants may deposit pheromone on the walks they performed. The algorithm stops if a certain criterion is fulfilled. This criterion may, for example, take the solution quality or a certain runtime into account.
More information on ACO can be found in the book "Ant Colony Optimization" by Dorigo and Stützle[36]. In the following paragraph, special features of the P-ACO algorithm are described.

In the multi-objective context, P-ACO extends ACO (i) by an additional outer iteration called *periods* in which random weights for each objective function are chosen, (ii) by checks whether a newly found solution is non-dominated by candidate solutions in a current solution set and vice versa, and (iii) by an objective-specific pheromone handling mechanism. For the selection of promising solutions in each step of the inner iteration (called *rounds*), P-ACO needs a scalarizing function. Different approaches (aggregation methods) can be used, e.g. weighted sums or a weighted Chebyshev distance function to an ideal point of the problem, i.e., a point each component of which is an upper bound of the corresponding objective function values (cf. Ralphs et al.[104]). The scalarization by weighted averages is a simple, intuitive approach to reduce multi-objective problems to single-objective ones; it assumes that the utility function of the decision maker is a linear function. A disadvantage of this type of scalarization is that not every Pareto-optimal solution can be represented as an optimal solution with respect to some weighted average. Optimization with respect to weighted Chebyshev distances (which is less intuitive) does not have this disadvantage, it can "reach" every Pareto-optimal solution.

The single objective functions $f^{(1)}(x), \ldots, f^{(p)}(x)$ are aggregated according to the chosen method. At the beginning of each period, the weights $w^{(1)}, \ldots, w^{(p)}$ are drawn randomly. In each period the solutions are gradually improved with respect to the current aggregated objective function $f(x)$.
A separate pheromone matrix $\tau^{(\vartheta)} = \left(\tau_{kl}^{(\vartheta)}\right)$ is assigned to each objective function $f^{(\vartheta)}$. The guiding pheromone matrix $\tau_{kl}$ is calculated as the weighted sum of objective specific pheromone values $\tau_{kl}^{(\vartheta)}$, using the weights $w^{(1)}, \ldots, w^{(p)}$. The main differences between the original algorithm and our implementation are, that in the original works each ant has each own weight vector to aggregate the individual pheromone matrices to guide the construction process, whereas our implementation follows an iterative weighted metric ap-

proach. All ants during one period share the same weight vector, therefore the each period corresponds to solving a single-objective optimization problem.

**Algorithm 2.4.3:** P-ACO

$\tau_{kl}^{(\vartheta)} = 1$ for all $(k,l)$ and for all $\vartheta = 1,\ldots,p$;
initialize the solution set $X$ as the empty set;
**for** *period* $\pi = 1,\ldots,\Pi$ **do**
  draw weights $w = \left(w^{(1)},\ldots,w^{(p)}\right)$ randomly;
  $\tau = \sum_{\vartheta=1}^{p} w^{(\vartheta)}\tau^{(\vartheta)}$;
  **for** *round* $m = 1,\ldots,M$ **do**
    **for** *ant* $\gamma = 1,\ldots,\Gamma$ **do**
      set $k$ equal to start node of $\mathcal{C}$;
      set $u$ equal to the empty set;
      **while** *a feasible continuation* $(k,l)$ *of* $u$ *exists* **do**
        select successor node $l$ with probability;

$$p_{kl} = \begin{cases} 0, \text{if}\,(k,l)\,\text{is infeasible} \\ \frac{[\tau_{kl}]^{\alpha}[\eta_{kl}(u)]^{\beta}}{\sum_{(k,r)}[\tau_{kr}]^{\alpha}[\eta_{kr}(u)]^{\beta}} \end{cases};$$

        the sum being over all feasible $(k,r)$ ;
        $k = l$, and append $l$ to $u$;
      **end**
      $x_\gamma = u$;
    **end**
    $f(x) = aggregate\left(f^{(1)},\ldots,f_{(p)};w\right)$;
    select the best walk $x$ out of $x_1,\ldots,x_\Gamma$;
    **if** $m = 1$ **then**
      $\hat{x} = x$
    **else**
      **if** $f(x) - f(\hat{x}) \leq 0$ **then**
        $\hat{x} = x$
      **end**
    **end**
    evaporation: $\tau^{(\vartheta)} = (1-\rho)\,\tau^{(\vartheta)}$ for all $\vartheta$;
    global-best reinforcement: $\tau_{kl}^{(\vartheta)} = \tau_{kl}^{(\vartheta)} + c_1 w^{(\vartheta)}$ for all $(k,l) \in \hat{x}$ and all $\vartheta$;
    round-best reinforcement: $\tau_{kl}^{(\vartheta)} = \tau_{kl}^{(\vartheta)} + c_2 w^{(\vartheta)}$ for all $(k,l) \in x$ and all $\vartheta$;
    $\tau = \sum_{r=1}^{R} w_r \tau^{(r)}$;
    **if** $\hat{x}$ *nondominated by* $X$ **then**
      add $\hat{x}$ to $X$ and remove dominated elements from $X$
    **end**
  **end**
**end**
**Output**: set of Pareto-optimal decision vectors $X$

### 2.4.4. Adaptive $\varepsilon$-Constraint Algorithm

As the traditional $\varepsilon$-constraint method (Haimes et al.[59]), also the adaptive $\varepsilon$-constraint method (Laumanns et al.[81]) works by choosing one of $m$ objectives of a multi-objective problem as the only objective and the remaining $m-1$ objective functions as constraints. For a bi-objective optimization problem where both objectives should be minimized, the constrained problem has the following form:

$$\begin{aligned} &\text{lex min } f(x) = (f_1(x), f_2(x)) \\ &\quad \text{s.t. } f_2(x) < \varepsilon_2, \\ &\qquad x \in X, \end{aligned} \tag{2.9}$$

where "lex min" denotes the lexicographic minimization of the two objectives.

In general, the "lex min" operator is needed to solve the technical complication that arises if solutions are possible that are *weakly Pareto-optimal* but not Pareto-optimal[2]. In the bi-objective case, however, the "lex min" operator is not needed, provided that an auxiliary procedure, eliminating weakly Pareto-optimal solutions that are not Pareto-optimal, is applied to the result set of the adaptive $\varepsilon$-constraint method. In this case, the objective function of (2.9) reduces to $f(x) = f_1(x)$. In the remainder of this work, we assume that such an auxiliary procedure is used, and always assume $f(x) = f_1(x)$.

Suppose we have a procedure $opt(f, \varepsilon_2)$ returning the optimal solution $x$ of the constrained problem or null if the problem is infeasible. Algorithm 3.1 shows the pseudocode of the adaptive $\varepsilon$-constraint method for a bi-objective minimization problem.

**Algorithm 2.4.4:** Bi-Objective Adaptive $\varepsilon$-Constraint Method

$P := \emptyset,\ \varepsilon_2 = \infty$;
**repeat**
  $\mathbf{x} := opt(\mathbf{f}, \varepsilon_2)$;
  **if** $\mathbf{x} \neq Null$ **then**
    $P := P \cup \{\mathbf{x}\}$;
    $\varepsilon_2 := f_2(\mathbf{x})$;
  **end**
**until** $\mathbf{x} = Null$;
**Output**: set of Pareto-optimal decision vectors $P$

When the algorithm starts, no bound for the second objective function is set. In each

[2] A solution $x^{(1)}$ is called weakly Pareto-optimal if there is no other solution $x^{(2)}$ that is strictly better than $x^{(1)}$ in all objectives.

iteration, the constrained single objective problem is solved. If a new solution is found, the solution is added to the Pareto set. The upper bound for the second objective function is set to the current objective function value. The algorithm stops as soon as the constrained single objective problem turns out as infeasible.

As the used MIP solver does not support inequalities of the form $f(x) < K$, such constraints must be replaced by inequalities of the form $f(x) \leq K - \Delta$. Therefore, in our implementation, the constraint $f_2(x) < \varepsilon_2$ is replaced by $f_2(x) \leq \varepsilon_2 - \Delta$.

# 3. Application to Project Portfolio Selection

## 3.1. Problem Description

To ensure their success, companies or organizations in competitive environments need to manage their resources such that they are used in the most effective way. Almost every organization, institution or company is faced with the question of *what* to do, and *how* to do what needs to be done, considering the limited resources. Managers need to identify the "best" subset of projects (a project portfolio) to be pursued among a sometimes large set of project proposals and set these up properly (schedule them and provide the necessary resources). Providing the necessary resources includes assigning employees with proper competencies to the selected projects. For various reasons these tasks are challenging: (i) selecting a subset of projects considering limited resources is a knapsack-type problem that is known to be $\mathcal{NP}$-hard, and (ii) to determine the feasibility of a given portfolio, the projects have to be scheduled and staff must be assigned to them, two (sub-) tasks that are difficult themselves. Furthermore the staffing decision determines the development of the employees' competence levels, which influences their ability to work in later projects. By implementing the selected projects, the assigned employees obtain new skills, that contribute to the (then extended) competence resources. The considered problem, is therefore characterized by a set of incomparable and conflicting objectives, that may be roughly classified as economic objectives and competence-oriented objectives. Economic objectives (e.g., return of investment), that are are functions of the project portfolio alone are common in literature. The explicit consideration of competence-oriented objectives is motivated by the fact that competencies (i.e., pragmatic knowledge in the sense of "know how") increasingly determine the attainment and sustainability of strategic positions in market competition. Companies may choose for a internal development of competencies for different reasons: (i) specific competencies are not easily available at any given time in the required quality/and or capacity on the (job) market, and (ii) integrating new employees in a established team of workers may be costly (e.g. communication and coordination efforts may increase). On the other hand, the in-house "production" of competencies may also be fairly costly and time-consuming. Considering the assets and drawbacks of in-house "production" of competencies, the decision on which competencies to develop to which degree is therefore of high managerial relevance. Considering both economic and competence-oriented objectives at the same time, yields a multi-objective optimization problem. As the objectives are incomparable and conflicting, the problem will have an efficient frontier consisting of several, pairwise incomparable (Pareto-optimal) solutions. In this work we present different methods to identify these solutions.

Identifying the set of Pareto-optimal solutions is a nontrivial task. The available methods for multi-criteria decision-making can be roughly classified by two complementary families: (i) mathematics-based multiple objective programming (MOP) and (ii) decision maker-driven multiple criteria decision analysis (MCDA). In this work we consider the MOP-part of the competence-oriented project portfolio selection problem, we develop a mathematical program as well as suitable solution procedures to identify Pareto-optimal solutions. The results of the developed solution procedures can then be used by an interactive system that incorporates the decision makers' judgments and preferences to identifying their individually "best" solution (which would constitute the MCDA part).

Another typically encountered aspect in practical project portfolio management is *uncertainty*. Benefits from projects can be uncertain, there can be the risk that a project for which a decision has been made will not come about, and, maybe most importantly, the amount of time and effort required to complete a project is often uncertain to a large extent. In this work, we will focus on the last type of uncertainty, but also include the first one into the model.

### 3.1.1. Related Literature

As already stated our model integrates *project portfolio selection* and *scheduling and staffing* decisions. These problems (and their combinations) have already found considerable interest, therefore numerous articles dealing with such problems exist. Articles that use linear, integer or dynamic programming techniques to support the single-objective portfolio selection decision appeared already in the 60s of the last century Asher [10], Beged-Dov [15], Hess [64]. Later complicating factors arising from practical applications were incorporated, adding more realism to the models. The use of mathematical models and software solvers to support the decision of managers, showed that decision makers are rarely willing to accept the "optimal" portfolio, but they are seeking for computer support to reduce the numerous number of numerous possible portfolios to a candidate set of reasonable portfolios. Decision maker can then evaluate and discuss these portfolios, and the final choice remains within their responsibility. *Multi-objective* project portfolio selection methods, such as goal programming (e.g.,Badri et al.[13], Khorramshahgol and Gousty[74]), scoring models (e.g., Henriksen and Traynor[63] or the Analytic Hierarchical Process (e.g., Gabriel et al.[47], Greenberg and Nunamaker[49], Suh et al.[115]), or techniques for the determination of Pareto-optimal solutions ,(e.g., Doerner et al.[34,35], Medaglia et al.[88], Stummer and Heidenberger[111], Stummer and Sun[112]) facilitate such decision processes.

More realism is added to project portfolio selection by taking the time structure and the personnel requirements of projects into account. Part of the literature emphasizes the scheduling view, that led to the development of models and solution procedures for the *resource-constrained project scheduling problem* Kolisch and Hartmann[76]. The other group of articles, where the staff assignment (the assignment of work packages of a project to employees or workers) view prevails, typically also include the question of varying skills (*competencies*) within the employees Alba and Francisco Chicano[2], Eiselt and Marianov[39], Gutjahr[52], Yoshimura et al.[126]. Examples for competence models are given in Mansfield[87], Shippmann et al.[108]. As in general, a person develops over time, competencies are not fixed, on the on hand competencies may be trained but also may deteriorate if not kept "up to date" on the other hand. This raises the question of *competence development*. Articles that treat competence development in quantitative models are e.g. Armbruster et al.[6], Chen and Edgington[25], Heimerl and Kolisch[61], Pendharkar and Subramanian[97], Suer and Tummaluri[114], Wu and Sun[125].

In practical project portfolio management, also aspects such as *uncertainty*, robustness or the dynamics of the portfolio selection exist. These aspects are investigated in e.g. Gabriel et al.[47], Kavadias and Loch[73], Liesiö et al.[83], Loch and Kavadias[84], Medaglia et al.[88]. Especially uncertainty is typically part of practical project portfolio management. Many parameters of project portfolio selection models can be uncertain, e.g. benefits can be uncertain, some projects that are included in the portfolio will not come about, and, maybe most importantly, the amount or resources (time, effort) to complete a project is often uncertain to a large extent. None of the mentioned articles, however, integrates portfolio selection, staff assignment, uncertainty and learning of competencies in a holistic model.

The remainder of this chapter essentially based on the articles Gutjahr and Reiter[55], Gutjahr et al.[58] and is organized as follows: Section 3.2 provides the formulation of the multi-objective mathematical programming model and provides linear asymptotic approximations as well as a description of the stochastic extension. Section 3.4 decomposes the problem into two subproblems and introduces solution procedures for the deterministic and stochastic model. Next, Section 3.5 describes the test instances that are used for the computational experiments. Section 3.6 describes experiments with synthetically-generated and real-life test cases to illustrate the performance of our solution techniques for the deterministic and stochastic models. Conclusions, as well as an outlook for further research, are presented in Section 3.7.

## 3.2. Model Formulation

Our multi-objective models are based on the single-objective *Project Selection, Scheduling and Staffing with Learning* problem (short: PSSSL problem) Gutjahr et al.[57]. In the following sections, we recapitulate the essential elements of the PSSSL model, and extend the model by adding multiple objective functions. We abbreviate the multi-objective model by MPSSSL. The considered problem belongs to the class of multi-objective mixed-integer optimization problems. iFurthermore we introduce the stochastic extension of the multi-objective problem that we abbreviate by SMPSSSL.

### 3.2.1. Project Portfolios

We assume that $n$ *project candidates* (opportunities) $i = 1, \dots, n$ are given. Each project consists of one or more *tasks*. We label the tasks by $k = 1, \dots, K$. The assignment of tasks to projects is given by constant indicator variables $c_{ik}$, where $c_{ik} = 1$ if project $i$ contains task $k$, and $c_{ik} = 0$ otherwise. Since each task belongs to exactly one project, the numbers $c_{ik}$ satisfy $\sum_{i=1}^{n} c_{ik} = 1$ for all $k$. Our aim is to provide decision support for the selection of a subset of projects, that is, a so-called *project portfolio*. The decision which candidate project is to be realized is represented by decision variables $y_i$ ($i = 1, \dots, n$), where $y_i$ takes the value 1 if project $i$ is included in the portfolio, and the value 0 otherwise. Note that we only allow 0-1-decisions about projects, that is, we do not consider the possibility of funding projects only partially.

A fixed time interval consisting of $T$ *periods* is considered. Periods are indexed by $t = 1, \dots, T$. (Typically, a period consists of one month.) Period $t$ starts at time $t - 1$ and ends at time $t$. We restrict ourselves to a *static* version of the decision problem where it is assumed that the decision on the projects to be selected has to be made at time $t = 0$, the start time of period 1, and remains invariant until the end of the time horizon (time $T$).

Furthermore, we assume that for each task $k$, the following numbers are given: (i) the *ready time* $\rho_k \in \{1, \dots, T\}$ of task $k$, and (ii) the *due date* $\delta_k \in \{1, \dots, T\}$ of task $k$. The ready time $\rho_k$ and the due date $\delta_k$ are defined as the first and the last period, respectively, where work in task $k$ is possible; that is, work in task $k$ can start not earlier than at time point $\rho_k - 1$ and must be completed not later than at time point $\delta_k$.

### 3.2.2. Employee Allocation

Our analysis will be carried out not on the aggregated level of the entire working team, but on the individual level of employees, which is much more realistic in several aspects (concerning work assignment and competence growth) than an aggregated consideration. The *employees* are indexed by $j = 1, \ldots, m$. It is assumed that the free working *capacity* of each employee $j$ in each period $t$ is given as the number $a_{jt}$ ($j = 1, \ldots, m$, $t = 1, \ldots, T$). We measure all work times in multiples of the standard work time in one period (e.g., the regular work time within one month in a full-time job). In particular, also the capacities $a_{jt}$ are expressed in multiples of this unit, such that they are typically numbers between 0 and 1. Also we suppose that the set of employees is fixed during the entire planning interval $[0, T]$. In other words, extensions of the staff, terminations of employment, outsourcing etc. are not taken into account.

The different fields of knowledge, education, skills etc. in which the employees can have abilities (relevant for the company) are called *competencies*. We index competencies by $r = 1, \ldots, R$. The degree to which an employee $j$ possesses a certain competency $r$ at time $t$ is quantified by a real (possibly also negative) value $z_{jrt}$ which we call the *competence score*. It is assumed that by learning, $z_{jrt}$ grows when employee $j$ works in a task requiring competency $r$, and that $z_{jrt}$ diminishes by the so-called knowledge depreciation effect when employee $j$ is not active in competency $r$. Initial values $z_{jr1}$ of the competency scores in period 1 are assumed as given. Methods for measuring competence scores will not be discussed in this paper; as to this subject, we refer to the literature on labor psychology.

The *efficiency* of employee $j$ in competency $r$, denoted by $\gamma_{jrt}$, is defined as the share of work performed in one time unit by employee $j$ on a task requiring only competency $r$, if the entire task takes one time unit for an employee with "perfect ability" in competency $r$ (cf., e.g., Wu and Sun[125]). An efficiency value of $\gamma_{rjt} = 1$ means that employee $j$ is "fully competent" in competency $r$ and will be able to perform parts of tasks that require this competency in the minimum possible time. If $0 < \gamma_{rjt} < 1$, we assume that employee $j$ is in principle able to work on a part of a task requiring competency $r$, but delivers per time unit only a share $\gamma_{rjt}$ of the performance of an employee with efficiency 1. Thus, work in competency $r$ that requires one period for an employee with efficiency 1 will take two periods if assigned to an employee with efficiency 0.5. We say that *real* work time of one time unit in competency $r$, invested by an employee with efficiency $\gamma_{rjt}$, will contribute to the completion of the task by an amount of *effective* work time of only $\gamma_{rjt}$ time units. Employees $j$ with efficiency $\gamma_{rjt} = 0$ in competency $r$ cannot contribute to parts of a task requiring this competency.

Although it can be expected that in general, the efficiency value will be an increasing function of the competence score, the exact functional relation depends on the way the competency score has been measured. For our purposes, we assume that $\gamma_{jrt}$ can be obtained from $z_{jrt}$ by applying some (in general non-linear) increasing function $\varphi_r$, which may depend on the specific competency $r$. The function $\varphi_r$ maps the set of reals into the interval $[0,1]$. [1] A viable approach to approximate $\varphi_r$ is to consider a parametrized class of functions suggested by theoretical considerations, and to estimate the parameters from empirical data. In the present paper, a logistic function (cf. Chen and Edgington[25], Ngwenyama et al.[93]) was utilized to transform the competence score to an efficiency value, i.e., we specified the function $\varphi_r(z)$ as $\varphi_r(z) = [1 + a\exp(-bz)]^{-1}$.

Task $k$ is assumed to require an overall *effective work time* of $d_{kr}$ in competency $r$ ($k = 1,\ldots,K$; $r = 1,\ldots,R$). The effective work time $d_{kr}$ is the time required by an employee with maximal efficiency $\gamma_{jrt} = 1$ for completing that part of the task that is related to competency $r$. We call this part the *work package* with index $(k,r)$. As the unit for work times, we take the overall maximum possible work time in one period. In the deterministic model of this work, the (real) numbers $d_{kr}$ are assumed as deterministic and known in advance.

In some cases, it is not realistic to assume that all work of a certain work package can be done "at once", but rather work has to be extended over a larger interval of time (an example are support activities). To be able to model this situation, we introduce upper bounds $b_{kr}$ for the expected effective work time invested per period into work package $(k,r)$. If there is no such bound for a work package, we set $b_{kr} = \infty$.

To describe (i) the scheduling of the selected projects over time with respect to their required work times, ready times and due dates, and (ii) the assignment of staff to the tasks of the selected projects with special attention given to the required competencies, we introduce real-valued decision variables $x_{kjrt} \in [0,1]$, where $x_{kjrt}$ denotes the *real* time employee $j$ works within period $t$ in competence $r$ of task $k$ ($k = 1,\ldots,K$; $j = 1,\ldots,m$; $r = 1,\ldots R$; $t = 1,\ldots,T$). As in the case of effective work times and capacities, time is again measured in multiples of the overall maximum possible work time in one period. The amount of effective work time contributed by real work time $x_{kjrt}$ of employee $j$ in competency $r$ and period $t$ is then given as $\gamma_{jrt}\, x_{kjrt}$. In total, the variables $x_{kjrt}$ form the (4-dimensional) *work time array* $x$.

[1] Our model allows negative values of the competence score. This does not mean, however, that the *efficiency* in the corresponding competence can fall below zero.

### 3.2.3. Competence Dynamics and Learning

To represent *learning*, the competence score of an employee $j$ in competency $r$ is assumed to increase in each period where employee $j$ has worked during an amount $x$ of (real) time in competency $r$ by an increment of size $\eta_r \cdot x$, where the factor $\eta_r$ is a constant that can depend on $r$. Similarly, we assume that the competency score of an employee $j$ in competency $r$ is reduced by the amount $\beta_r$ in each time period by knowledge depreciation. Evidently, this loss can be over-compensated by the gain achieved by activity in competency $r$ provided that $\beta_r < \eta_r$. The parameters $\eta_r$ and $\beta_r$ are called the *learning rate* and the *depreciation rate* of competency $r$, respectively.

### 3.2.4. Objective Functions

For defining our objective functions, quantities describing the *gains* from projects are needed. We distinguish two classes of gains:

*(i) Economic gains* such as return, turnover etc. Whatever types of economic gains are chosen for formulating the objectives, it can be assumed that gains are assigned to projects, more specifically, that they result from the completion of projects that have been included in the portfolio. By $w^{(\pi)} = (w_1^{(\pi)}, \ldots, w_n^{(\pi)})$ $(\pi = 1, \ldots, p)$, we denote the *economic benefits* resulting from the inclusion of project $i$ in the portfolio $(i = 1, \ldots, n)$. For example, $w_i^{(1)}$ can measure profit contribution and $w_i^{(2)}$ turnover contribution, respectively, achieved from project $i$ $(i = 1, \ldots, n)$. Often, there are positive or negative interactions between different projects (called *synergy* resp. *cannibalization* effects): If two projects $i$ and $s$ are included in the portfolio, their common gain can exceed $w_i^{(\pi)} + w_j^{(\pi)}$ or fall below this sum. We take account of this phenomenon by adding, for each pair $(i, j)$ of projects contained in the portfolio, a corresponding term $w_{ij}^{(\pi)}$ that can be positive, negative or zero, to the overall gain. [2]

*(ii) Strategic gains.* They result from the strategic development of the organization or firm into desirable directions, taking probable future changes in the market situation into account. It is important to include gains of this type in the model as well, since otherwise, optimization would exclusively concentrate on short-term financial gains, neglecting the long-term competitiveness of the firm. The tradeoff between short-term and long-term goals is well-known in the strategic management literature and should be addressed by an adequate model. In order to formulate strategic goals quantitatively, we build on the list of competencies and assume that for $\kappa = 1, \ldots, q$, vectors $v^{(\kappa)} = (v_1^{(\kappa)}, \ldots, v_R^{(\kappa)})$

[2] Another way to deal with synergy and cannibalization would be to introduce dummy projects (cf. Liesiö et al. [83]).

of *competence weights* for competencies 1 to $R$ are given. Each vector $v^{(\kappa)}$ represents a (desired) *competence profile*. Different competence profiles [3] can reflect different strategic viewpoints or aims of different stakeholders. (Allowing multiple weight vectors is in agreement with classical approaches in decision analysis, cf. Arbel [4], Weber [124].) The $\kappa$-th strategy weight $v_r^{(\kappa)}$ quantifies the importance of competency $r$ with respect to competence profile $\kappa$. Engaging in projects that involve competencies with high $v_r^{(\kappa)}$ makes the firm more competitive in future market scenarios, whereas investing into projects that involve competencies with low $v_r^{(\kappa)}$ (i.e., competencies that may become obsolete) may result in short-term profits, but at the cost of long-term stability. The degree of engagement in a competency is measured by the total amount of expected (real) work time invested during the planning interval into work packages assigned to this competency. The competence-oriented objectives refer to the end of the planning horizon, that is, the situation in (the beginning of) period $T+1$. In the multi-objective decision approach, it is not necessary that the competence weights $v_r^{(\kappa)}$ are scaled in a specific way in relation to the economic benefits $w_i^{(\pi)}$.

### 3.2.5. Mathematical Programming Formulation

For the arrays $x = (x_{kjrt})$ and $y = (y_i)$ of decision variables, we define two sets of objective functions:

$$f^{(\pi)}(y) = \sum_{i=1}^{n} w_i^{(\pi)} y_i + \sum_{i<j} w_{ij}^{(\pi)} y_i y_j \qquad (\pi = 1, \ldots, p) \tag{3.1}$$

and

$$g^{(\kappa)}(x) = \sum_{r=1}^{R} v_r^{(\kappa)} \sum_{j=1}^{m} (\gamma_{j,r,T+1} - \gamma_{jr1}) \ (\kappa = 1, \ldots, q). \tag{3.2}$$

The first set (3.1) of objective functions represents the economic benefits from the selected projects. The objective function $f^{(\pi)}(y)$ in this set measures the economic benefit according to the values $w_i^{(\pi)}$ assigned to the single projects. The second set (3.2) of objective functions represents the competence benefits obtained from the increments of the efficiencies $\gamma_{jrt}$ over the planning horizon. The objective function $g^{(\kappa)}(x)$ in this set measures the total increment of weighted efficiencies, cumulated over employees, where the efficiency value corresponding to competency $r$ is weighted by the importance value $v_r^{(\kappa)}$. Contrary

[3] The weights $v_r^{(\kappa)}$ for each competency $r$ of a competence profile $\kappa$ indicate to which degree this competency will be required in the *future* market situation.

to (3.1), the objective functions (3.2) do not depend on the portfolio decision vector $y$ in a direct way, but only indirectly via the work time array $x$.
The problem MPSSSL is then defined as follows:

$$\text{(MPSSSL)} \max \left(f^{(1)}(y), \ldots, f^{(p)}(y), g^{(1)}(x), \ldots, g^{(q)}(x)\right) \tag{3.3}$$

s.t. (3.1), (3.2) and

$$\gamma_{jrt} = \varphi_r(z_{jrt}) \tag{3.4}$$

$$z_{jrt} = z_{jr1} - \beta_r(t-1) + \eta_r \sum_{k=1}^{K} \sum_{s=1}^{t-1} x_{kjrs} \quad \forall j, r, t \tag{3.5}$$

$$\sum_{k=1}^{K} \sum_{r=1}^{R} x_{kjrt} \leq a_{jt} \quad \forall j, t \tag{3.6}$$

$$\sum_{t=\rho_k}^{\delta_k} \sum_{j=1}^{m} \gamma_{jrt} x_{kjrt} = d_{kr} \sum_{i=1}^{n} c_{ik} y_i \quad \forall k, r \tag{3.7}$$

$$\sum_{j=1}^{m} \gamma_{jrt} x_{kjrt} \leq b_{kr} \quad \forall k, r, t \tag{3.8}$$

$$(t - \rho_k) x_{kjrt} \geq 0 \text{ and } (\delta_k - t) x_{kjrt} \geq 0 \quad \forall k, j, r, t \tag{3.9}$$

$$x_{kjrt} \geq 0 \quad \forall k, j, r, t, \tag{3.10}$$

$$y_i \in \{0, 1\} \quad \forall i. \tag{3.11}$$

Constraints (3.4) specify the dependence of the efficiency values $\gamma_{jrt}$ on the competence scores $z_{jrt}$. Constraints (3.5) represent the evolution of the competence scores by knowledge depreciation and by learning. Note that we assume that the competence score remains fixed *during* a period, which is only a valid approximation if the period length is chosen as comparably short. Constraints (3.6) bound the invested real work time of each employee by her or his capacity limit. Constraints (3.7) ensure that the real work time of each employee in a competency $r$ within a given task $k$, multiplied by her or his efficiency (which gives the effective work time), and cumulated over all employees and over the runtime of the task, must yield the overall required effective work time $d_{kr}$ for work package $(k, r)$, if the project to which task $k$ belongs is selected in the portfolio, and zero otherwise. Constraints (3.8) bound the effective work times in each work package by the maximum allowed amount per period. Constraints (3.9) ensure that before the ready time and after the due date of a task, no work is spent to this task. Constraints (3.10) restrict the decision variables to their allowed ranges.[4]

As it can be seen, the formulation above allows it to distinguish different categories of economic gains and to represent them as different objectives. Similarly, different "strategic lines" for competence development, each represented by a weight vector $v^{(\kappa)}$, can be taken into account in the form of separate objectives.

The elicitation of the weights $v_r^{(\kappa)}$ may be supported by a variety of methods (for an overview, cf. Belton and Stewart [17]). A natural choice would be to rely on the preference comparison methods of Multiattribute Utility Theory (MAUT), applying an additive model which is generally quite robust Butler et al. [22] and is consistent with our linear problem formulation. Also applying the Analytical Hierarchy Process (for an example, cf. Arbel [4]) may be a promising approach. When eliciting the weights, the assessment of each vector $v^{(\kappa)}$ can be based on a separate stakeholder group.

Even in the special case where the functions $\varphi_r$ are linear, the MPSSSL problem is a non-linear multi-objective problem: The variables $\gamma_{jrt}$, which depend on the decision variables $x_{kjrt}$ by (3.4) – (3.5), are multiplied by the variables $x_{kjrt}$ in (3.7).

### 3.2.6. Pareto-optimal Solutions

Because of the multi-objective nature of our problem formulation, the decision maker cannot be provided with a single "optimal" solution. Instead, we determine (an approximation to) the set of *Pareto-optimal solutions*. The decision variable, that is, the "solution" to the problem, is denoted by $u$ in the definitions below. In the case of our problem MPSSSL, $u$ is given by the pair $(y, x)$, where $y$ is the (binary) project portfolio vector, and $x$ is the (real-valued) work time array. The objective functions (to be maximized) are written as $\Psi_1, \ldots, \Psi_D$ below; in the MPSSSL case, these $D$ objective functions consist of the two groups $f^{(1)}, \ldots, f^{(p)}$ and $g^{(1)}, \ldots, g^{(q)}$, such that $D = p + q$. As seen from Gutjahr et al. [57], already the single-objective version PSSSL of the MPSSSL problem is hard to solve, which is not surprising in view of the nonlinear and mixed-integer problem characteristics. For this reason, it cannot be expected that real-world instances of the MPSSSL problem can be solved exactly within reasonable computation time. Instead, we shall propose the application of multi-objective metaheuristics in order to obtain suitable approximations to the set of Pareto-optimal solutions.

[4] By a very small extension of the model, also *formal training* of employees, e.g., courses where specific skills are acquired, could be represented. We omit the details for the sake of brevity.

### 3.2.7. Linear Asymptotic Approximation

To transform the originally nonlinear problem formulation (3.1) – (3.10) into a linear one, which is somewhat easier to solve, we assume that usually, both learning rates $\eta_r$ and depreciation rates $\beta_r$ are small compared to unity.[5] Even in cases of not very small rates $\eta_r$ and $\beta_r$, the solution obtained by the asymptotic linearization might be used as an initial solution for a local search where the decision is fine-tuned in the nonlinear context. For the single-objective case, a corresponding asymptotic approximation has been presented in Gutjahr et al.[57], and the multi-objective situation is described in Gutjahr et al.[58]. Therefore we keep our presentation short and refer the reader to Gutjahr et al.[57,58] for technical details.

Mathematically, the assumption of small learning rates $\eta_r$ and small depreciation rates $\beta_r$ can be represented by setting

$$\eta_r = \bar{\eta}_r \cdot \epsilon \text{ and } \beta_r = \bar{\beta}_r \cdot \epsilon, \tag{3.12}$$

where $\bar{\eta}_r$ and $\bar{\beta}_r$ are constants, and $\epsilon \ll 1$. By combining (3.4) and (3.5) to a single equation and inserting (3.12), we obtain

$$\gamma_{jrt} = \gamma_{jrt}(\epsilon) = \varphi_r \left( z_{jr1} - \bar{\beta}_r \epsilon (t-1) + \bar{\eta}_r \epsilon \sum_{k=1}^{K} \sum_{s=1}^{t-1} x_{kjrs} \right) = \varphi_r \left( z_{jr1} + \epsilon h_{jrt} \right)$$

where

$$h_{jrt} = -\bar{\beta}_r (t-1) + \bar{\eta}_r \sum_{k=1}^{K} \sum_{s=1}^{t-1} x_{kjrs}.$$

By Taylor expansion at $\epsilon = 0$, we get

$$\gamma_{jrt} = \varphi_r(z_{jr1}) + h_{jrt} \cdot \varphi'(z_{jr1}) \cdot \epsilon + \frac{(h_{jrt})^2}{2} \cdot \varphi''(z_{jr1}) \cdot \epsilon^2 + O(\epsilon^3).$$

In a first-order approximation, we neglect already terms of order $O(\epsilon^2)$, such that

$$\gamma_{jrt}(\epsilon) \sim \varphi_r(z_{jr1}) + h_{jrt} \cdot \varphi'(z_{jr1}) \cdot \epsilon. \tag{3.13}$$

First of all, note that the objective functions $f^{(\pi)}(y)$ in (3.1) do not depend on $\epsilon$ (and

[5] Note that the typical length of a period is one month. Over the entire planning horizon $T$, small increments/decrements by learning may nevertheless cumulate to crucial differences. This does not hold anymore for terms that are small of *second order*, i.e., the $O(\epsilon^2)$ terms in the expansion below.

are already linear), so nothing has to be done there.
The objective function $g^{(\kappa)}(x)$ in (3.2) can be approximated by

$$\sum_{r=1}^{R} v_r^{(\kappa)} \sum_{j=1}^{m} (h_{j,r,T+1} - h_{jr1})\, \varphi_r'(z_{jr1})\, \epsilon = \epsilon \cdot \sum_{r=1}^{R} v_r^{(\kappa)} \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \left\{ -\bar{\beta}_r T + \bar{\eta}_r \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \right\}.$$

It is easy to see that by transforming an objective function in a multi-objective optimization problem by an increasing transformation function, the set of Pareto-optimal solutions does not change, since the dominance relations between solutions remain invariant. Observe that $\epsilon$, $v_r^{(\kappa)}$, $\varphi_r'(z_{jr1})$ and $\bar{\beta}_r T$ do not depend on the decision. Therefore, instead of maximizing the expression approximating $g^{(\kappa)}(x)$ above, one can also maximize the expression obtained by dividing the original expression by $\epsilon > 0$ and by adding then the constant $\sum_{r=1}^{R} v_r^{(\kappa)} \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \cdot \bar{\beta}_r T$. This yields the following transform of the approximated objective function:

$$\bar{g}^{(\kappa)}(x) = \sum_{r=1}^{R} v_r^{(\kappa)} \bar{\eta}_r \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \ (\kappa = 1, \ldots, q) \tag{3.14}$$

Let us now consider the constraints. We have already dealt with (3.4) – (3.5). Constraint (3.6) is linear. There remain constraints (3.7) – (3.8), containing the efficiencies $\gamma_{jrt}$. From (3.13) we see that a first order-approximation for $\sum_{j=1}^{m} \gamma_{jrt} x_{kjrt}$ is given by

$$\sum_{j=1}^{m} \gamma_{jrt} x_{kjrt} \sim \sum_{j=1}^{m} \varphi_r(z_{jr1}) x_{kjrt} = \sum_{j=1}^{m} \gamma_{jr1} x_{kjrt}. \tag{3.15}$$

Considering also the $O(\epsilon)$ approximation term from (3.13) in (3.7) or (3.8) would cause an influence of order $O(\epsilon)$ on the variables $x_{kjrt}$, which would add a correction term to the objective function (3.14) that is already negligible compared to the main term. Hence, also the approximated constraints are linear in the decision variables $x_{kjrt}$, and the first-order approximation problem of MPSSSL (LMPSSSL) is then defined as follows:

$$\text{(LMPSSSL)}\ \max\ \Big(f^{(1)}(y),\ldots,f^{(p)}(y),\bar{g}^{(1)}(x),\ldots,\bar{g}^{(q)}(x)\Big) \tag{3.16}$$

s.t. (3.1), (3.14), (3.7) and

$$\sum_{t=\rho_k}^{\delta_k}\sum_{j=1}^{m}\gamma_{jr1}x_{kjrt} = d_{kr}\sum_{i=1}^{n}c_{ik}y_i \qquad \forall k,r \tag{3.17}$$

$$\sum_{j=1}^{m}\sum_{j=1}^{m}\gamma_{jr1}x_{kjrt} \le b_{kr} \qquad \forall k,r,t \tag{3.18}$$

(3.9), (3.10), (3.11)

## 3.3. Stochastic Extension

As mentioned in Section 3.1 some articles models for portfolio selection incorporate *uncertainty*. Therefore we present in this section a stochastic extension of our deterministic multi-objective model described in 3.2.

We will use an additional objective function that measures the *robustness* of the portfolio by capturing expected surplus costs due to overtime or external work. We assume that the first set of objective functions (measuring economic and strategic gains) are deterministic whereas the robustness objective is given as the expected value of a random quantity. In the following section we start with the description of the generalization of the stochastic model formulated in Gutjahr and Reiter[55], taking into account several economic and/or strategic objective functions.

### 3.3.1. Stochastic Model Formulation

In Section 3.2.2 we denote a part of task $k$ that requires competency $r$ by the *work package* $(k,r)$ $(k = 1,\ldots,K,\ r = 1,\ldots,R)$. In addition to the deterministic model we now assume that the effective work time which is required by a certain work package $(k,r)$ is subject to uncertainty. Therefore we model this amount as a random variable with known mean $d_{kr}$. We suppose that the random fluctuations around $d_{kr}$ are project-specific. This can be expressed by introducing random variables $U_i$ $(i = 1,\ldots,n)$ corresponding to the projects, and assuming that the (random) work time in a work package $(k,r)$ assigned to project $i$ is given by $U_i\, d_{kr}$. For example, if project $i$ requires by 20 percent more time than estimated in advance, then the random variable $U_i$ takes the value 1.2, and the additional 20 percent of required effective work time are assumed to distribute proportionally over all work packages of which project $i$ consists. The random variables $U_i$ can be independent or dependent.

If it turns out that the required amount of work in a project $i$ has been underestimated, we suppose that the manager sticks nevertheless to the original work plan, and that the needed additional work is provided – as far as necessary – by *overtime*, i.e., by work exceeding the regular capacities $a_{jt}$ of the employees.[6] This is of course an essential restriction, since it excludes the possibilities of shifting the due dates of tasks or of allowing tardiness. The assumption is certainly adequate for branches of business where a "just-in-time" philosophy has been established, such that due dates are always "hard", but we think that it can also serve as an approximation for cases of soft due dates.

As mentioned in Section 3.3.1, we aim at judging the robustness of project plans with respect to wrong effort estimates, and the expected overtime cost can serve as a measure of (un-)robustness also in cases where it is possible to be tardy. In order to define the measure quantitatively, we consider numbers $g_j$ as given, where $g_j$ represents the wage per time unit for overtime of employee $j$ ($j = 1, \ldots, m$).

More specifically, we assume that the additional workload resulting from estimation errors is distributed over the employees and periods in proportion to the planned workload assignments. This assumption only approximates reality, since often the need for extra work becomes known only gradually during the execution of the project. However, for a rough estimate, the assumption makes sense, since overtime underestimation for a late stage of one project may be compensated by overtime overestimation for an early stage of another project, to which the same employee is assigned, such that, to some degree, the overtime estimation error is averaged over the periods.

As mentioned in Section 3.1 we also assume that the economic gains can be subject considerable uncertainty, partially caused by lacking information on the availability of buyers or customers, but also stemming from other sources. We take account of the uncertainty by treating $w_i^{(\pi)}$ (economic gains) and $w_{ij}^{(\pi)}$ (interaction effect) as random variables as well, and assume their distributions to be known. In this work, we suppose the decision maker to be risk-neutral in the sense that s/he aims at maximizing the expected value of the gain, neglecting higher moments of the gain distribution. The expected

[6]Two alternative assumptions are to suppose that in this case, (i) external work is used, i.e., parts of the work are outsourced, or (ii) capacity overflows are handled by means of internal re-assignments of employees. We shall discuss these alternatives at the end of this subsection.

economic gain can be expressed as

$$\begin{aligned} f^{(\pi)}(y) &= \mathbb{E}\left(\sum_{i=1}^{n} w_i^{(\pi)} y_i + \sum_{i<j} w_{ij}^{(\pi)} y_i y_j\right) \\ &= \sum_{i=1}^{n} \mathbb{E}\left(w_i^{(\pi)}\right) y_i + \sum_{i<j} \mathbb{E}\left(w_{ij}^{(\pi)}\right) y_i y_j \ (\pi = 1, \ldots, p) \end{aligned} \tag{3.19}$$

The numbers $\mathbb{E}(w_i^{(\pi)})$ $(i = 1, \ldots, n)$ and $\mathbb{E}(w_{ij}^{(\pi)})$ $(1 \leq i < j \leq n)$ are $n(n+1)p/2$ parameters that have to be estimated. (In the absence of synergies and cannibalization effects, it suffices to estimate the $np$ expected gains $\mathbb{E}(w_i^{(\pi))}$.)

Using these assumptions, a first version of our multi-objective stochastic optimization problem can be formulated as follows:

$$\text{(SMPSSSL)} \ \max \ \left(f^{(1)}(y), \ldots, f^{(p)}(y), g^{(1)}(x), \ldots, g^{(q)}(x), h(x)\right) \tag{3.20}$$

s.t. (3.2), (3.19), and

$$h(x) = -\mathbb{E}\left(\sum_{j=1}^{m} g_j \sum_{t=1}^{T} \left[\sum_{k=1}^{K} \sum_{r=1}^{R} \sum_{i=1}^{n} c_{ik} U_i \, x_{krjt} - a_{jt}\right]^{+}\right) \tag{3.21}$$

$$(3.4) - (3.11) \tag{3.22}$$

Constraints (3.2), (3.19) and (3.21) define the objective functions of the multi-objective problem. Constraints (3.19) are the expected economic gains from the selected projects. (3.2) represents the expected strategic gain: Note that $\sum_{k=1}^{K} \sum_{j=1}^{m} \sum_{t=1}^{T} x_{krjt}$ is the expected real work time invested into work belonging to competency $r$. It is important to observe that objective functions $f^{(\pi)}(y)$, $\pi = 1, \ldots, p$ are in fact deterministic, since $\mathbb{E}(w_i^{(\pi)})$ and $\mathbb{E}(w_{ij}^{(\pi)})$ are parameters that can be estimated in advance.

Constraint (3.21) defines the expected total overtime cost (the negative sign has been introduced in order to *maximize* with respect to both objective functions): With $i(k)$ denoting the project to which task $k$ is assigned, observe that replacing the planned work time $x_{krjt}$ by $U_{i(k)} x_{krjt} = \sum_{i=1}^{n} c_{ik} \, U_i \, x_{krjt}$, i.e., distributing the actual workload proportionally to the planned workload, yields an overtime for employee $j$ in period $t$ equal to the expression $[\ldots]^+$ in (3.21). Summation over all periods, multiplication by $g_j$ (overtime wage for employee $j$ per time unit) and summation over all employees gives the total overtime cost. It should be observed that our model assumes fixed *basic* personnel costs (which need not to enter into the model, because they form a decision-independent constant) irrespectively of whether the assigned workload requires the entire *regular* work time or whether the workload can be covered in a shorter time; thus, "undertime" has no effect, but only overtime leads to an additional cost term. Therefore, only the positive part of the difference between work time and capacity enters into objective function $h(x)$.

As the constraints of the stochastic model (SMPSSSL), are basically the same as in the deterministic model (MPSSSL) the stochastic situation can be treated similarly; thus we keep our presentation short and focus on the main differences. The main difference is that we assume that the constraints of the SMPSSSL model have to be fulfilled in the in the expected situation (i.e., the situation described by the replacement of all random variables by their expectations). Therefore constraint (3.7) ensures that in the expected situation, the required effort for each work package is covered by the work plan, and constraint (3.6) ensures that in the expected situation, the capacities of the employees are sufficient for the work plan to be executable without overtime. Constraint (3.7) also guarantees coverage of the required effort for each *stochastic* scenario: By our policy described above, we replace $x_{krjt}$ by $U_{i(k)}x_{krjt}$ in the stochastic case. If the last expression is inserted instead of $x_{krjt}$ on the left hand side of (3.7), we obtain, assuming that (3.7) is satisfied:

$$\sum_{t=\rho_k}^{\delta_k} \sum_{j=1}^{m} \gamma_{rj} \, U_{i(k)} \, x_{krjt} = U_{i(k)} \sum_{t=\rho_k}^{\delta_k} \sum_{j=1}^{m} \gamma_{rj} \, x_{krjt} = U_{i(k)} \, d_{kr} \sum_{i=1}^{n} c_{ik} y_i = U_{i(k)} \, d_{kr} y_{i(k)},$$

and the last expression just gives the required work time for work package $(k, r)$ in the stochastic situation, provided that the portfolio contains project $i(k)$, and zero otherwise. Constraint (3.8) ensures that the upper bounds $b_{kr}$ for the expected effective work time invested per period into work package $(k, r)$ are respected. Constraints (3.9) – (3.11) ensure that work in task $k$ before its ready date and after its due date is excluded, work times $x_{krjt}$ have to be nonnegative reals, and the decision variables $y_i$ on portfolios are binary integers.

Problem (3.20) – (3.22) is a mixed-integer, nonlinear, multi-objective stochastic program with feasible set (decision space) $\{0, 1\}^n \times \mathbb{R}^{KRmT}$.

Note that in the model above, the evaluation of the stochastic objective function $h(x)$ cannot be isolated from the employee-task-assignment problem, although uncertainty is only associated with entire projects. One might guess that it would be enough to estimate the total work time that each project needs in average and to make these figures random in order to estimate overtime. This is possible indeed for the total *effective* work time required for a set of selected projects. However, in order to compute $h(x)$, we need the *real* work time required by each employee instead, and real work time (which is related to effective work time by the factor $\gamma_{jr}$) depends on the employee-task assignment. Therefore, already in the deterministic boundary case where all $U_i$ are equal to unity, the employee-task assignment problem has to be solved. The presence of noise cannot dispense us from the additional complexity introduced by the varying competencies of employees.

As stated before the solution concept for multi-objective optimization problems used in this work is that of Pareto-optimal solutions. For (3.20) – (3.22), the efficient frontier is a continuous curve in $\mathbb{R}^2$. The combination of non-linearity, stochasticity and mixed-integer decision variables makes this problem computationally difficult to solve, at least for instances of practically relevant size (note that the number $K \cdot R \cdot m \cdot T$ of continuous decision variables is typically very large!). For this reason, we will consider a simplified model in the next subsection.

**Subcontractors:** Let us shortly discuss possible extensions of the model (3.20) – (3.22) to the case where part of the workload can be outsourced to subcontractors or to external personnel. We distinguish two essentially different situations: (i) Already at the beginning of the planning interval, certain activities or sub-projects are outsourced to subcontractors in order to reduce the risk of overtime in advance. (ii) Outsourcing takes place if and when it becomes necessary, i.e., if it turns out that in a certain period $t$, one or several employees are not able to cope with their workload, freelancers with the same qualification are searched on the labor market and paid for performing the necessary extra work on the basis of a short-term contract. Situation (i) can be dealt with by a very simple extension of our model: Assume that by a possible subcontract with another firm, part or all of the activities required for project $i$ can be outsourced, such that from the viewpoint of internal work, the project is reduced to a project $i'$ with decreased efforts $d'_{kr} \leq d_{kr}$ for its work packages. The compensation $p^{(c)}$ to be paid to the subcontractor reduces e.g. the expected profit $\pi = 0$ of project $i$ from $\mathbb{E}(w_i^{(0)})$ to $\mathbb{E}(w_{i'}^{(0)})$ with $w_{i'}^{(0)} = w_i^{(0)} - p^{(c)}$. Now, it suffices to include both project $i$ and project $i'$ in the problem instance and to add the additional constraint $y_i + y_{i'} \leq 1$ ensuring that only one of the two alternative projects (or none of them) is selected. Note that in the case of a subcontract closed in advance, the subcontractor bears the risk of possibly overtime in his/her part of the work, such that the real effort required for this part does not matter. Also situation (ii) can be treated by an extension of our model. Suppose that the competence profiles of employees can be classified into certain qualification types, such that employees $j$ of the same qualification type have the same efficiencies $\gamma_{rj}$ in each competency $r$. Then, in the case where in a certain period $t$, the capacity $a_{jt}$ of employee $j$ is exceeded by the actual workload $L$, one may look for some external worker $\ell = \ell(j)$ of the same qualification type on the labor market to employ her/him for the remainder of the work $(L - a_{jt})^+$. Of course, this is only advantageous if the regular wage $\bar{g}_\ell$ per hour of the stepping-in worker $\ell$ is smaller than the *overtime* wage $g_j$ per hour of employee $j$. All we have to change in the model is to replace in (3.21) the coefficients $g_j$ with $\tilde{g}_j = \min(g_j, \bar{g}_{\ell(j)})$. Often, however, the decision maker cannot be *sure* about the availability of external workers at the time when they will be needed.[7] It is easily possible to represent uncertainty on the availability of external workers within the model. We show this for the simpler case where a substitute for employee $j$ is either available in all required periods or in none; by a slight modification, also the case where availability depends on the period can be treated. Let $V_j$ the indicator variable for the random event that a substitute $\ell(j)$ for employee $j$ will be available. We replace the coefficients $\tilde{g}_j$ introduced above by $V_j\tilde{g}_j + (1-V_j)g_j$. Assuming that the event of availability of a substitute is independent of the random work time factors $U_i$ (which is a reasonable assumption for most cases), we can apply the product rule for independent random variables and rewrite the extended constraint (3.21) as

$$h\,(x) = -\mathbb{E}\left(\sum_{j=1}^{m}(\eta_j\tilde{g}_j + (1-\eta_j)g_j)\;\sum_{t=1}^{T}\left[\sum_{k=1}^{K}\sum_{r=1}^{R}\sum_{i=1}^{n} c_{ik}\,U_i\,x_{kjrt} - a_{jt}\right]^{+}\right), \qquad (3.23)$$

where $\eta_j$ is the probability that a substitute for $j$ will be available.

**Employee Re-Assignments:** A second possible model extension considers re-assignments of employees to tasks as soon as capacity bottlenecks become known during the execution of the projects. This alternative necessitates a *dynamic* planning process which is computationally much more complex than the static planning considered in our basic model. Although the main features of the model could be preserved in such an extension, the solution space would have to be enlarged from static decisions to a very large set of dynamic policies. We consider such an extension as a topic of future research (see Conclusions). In practice, dynamic personnel re-assignment is often considered as undesirable for accountability reasons, since repeated re-assignment of employees to different work packages tends to make a transparent assessment of individual contributions very difficult. Moreover, the familarization of an employee with a new task takes extra time and can even slow down the completion of the task (cf. Brooks Jr[21]), such that this option has to be used with much caution. (Clearly, the same limitation also holds for *ad hoc* subcontracting.) Therefore, already the static model makes sense from the viewpoint of applications. Nevertheless, also in cases where the decision maker uses the possibility of dynamic re-assignments whenever

[7]This does not hold for the availability of subcontractors in situation (i), because whether or not a subcontract can be made in advance can be judged immediately at the time of the planning decision.

it is advantageous, our model in its present form is still applicable: one has only to change the interpretation of objective function $h(x)$ from "expected total overtime cost" to a conservative estimate of this cost.

### Linear Asymptotic Approximation

Analogously to the linear asymptotic approximation of the deterministic model described in Section 3.2.7 a similar model can be obtained for the stochastic model. Which can be formulated as follows:

$$\text{(LSMPSSSL)} \max \left(f^{(1)}(y), \ldots, f^{(p)}(y), g^{(1)}(x), \ldots, g^{(q)}(x), h(x)\right) \tag{3.24}$$
$$\text{s.t. (3.2), (3.19), (3.21) and}$$
$$\text{(3.17), (3.18), (3.9), (3.10) (3.11)}$$

## 3.4. Solution Techniques

The MPSSSL problem as well as the SMPSSL problem admits a natural decomposition into two subproblems: The *master problem* consists in the portfolio selection, i.e., in the choice of the binary vector $y$. The *slave problem* that consists in the scheduling-and-staff-assignment decision, i.e., in the choice of the work time array $x$, given a fixed portfolio $y$. But the decompositions are slightly different for the deterministic respectively stochastic case. Thus we give a brief description of the two decomposition approaches in the following paragraphs.

**(i) Deterministic problem** In the deterministic case the master problem (MP) is a discrete multi-objective optimization problem with the set $\{0,1\}^n$ of binary vectors of length $n$ as the search space, and $p+q$ objectives.

$$\text{(MPdet)} \max \left(f^{(1)}(y), \ldots, f^{(p)}(y)\right) \tag{3.25}$$
$$\text{s.t. } f^{(\pi)}(y) = \sum_{i=1}^{n} w_i^{(\pi)} y_i + \sum_{i<j} w_{ij}^{(\pi)} y_i y_j \;(\pi = 1, \ldots, p) \tag{3.26}$$
$$y_i \in \{0,1\} \qquad \forall i. \tag{3.27}$$

Considering a special *fixed* portfolio $y$, the values of these $p+q$ objectives are (in general) not yet completely determined; instead, the solution of the slave problem described below assigns to the given $y$ a (possibly empty) set of solutions corresponding to a set of points in the objective space.

In general contrary to the master problem, the slave problem (SP) is a *continuous* multi-objective optimization problem. Its search space consists of the feasible work time arrays $x$ for the given fixed portfolio $y$ (this search space can also be empty). Since for fixed $y$, the first $p$ objective functions in (3.3) become fixed, the slave problem has actually only $q$ objectives.

If the linear approximations of Subsection 3.2.7 are applied, the slave problem reduces to a multi-objective *Linear Program* (LP).

$$\text{(SPdet)}\ \max\ \left(\bar{g}^{(1)}(x), \ldots, \bar{g}^{(q)}(x)\right) \tag{3.28}$$

$$\text{s.t. } \bar{g}^{(\kappa)}(x) = \sum_{r=1}^{R} v_r^{(\kappa)} \bar{\eta}_r \sum_{j=1}^{m} \varphi'_r(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs}\ (\kappa = 1, \ldots, q) \tag{3.29}$$

$$(3.9) – (3.11),\ (3.17),\ (3.18) \tag{3.30}$$

We shall focus on the special case $q = 1$ where the slave problem has actually only one objective, such that it consists in the solution of an ordinary (single-objective) LP.

The case $q > 1$ is considerably harder to treat computationally, since in this case, Pareto-optimal solutions are of mixed-integer type. Note that to each portfolio $y$ that occurs in the set of Pareto-optimal solutions, the solution of the slave problem consists of a composition of $(q-1)$-dimensional facets in the $q$-dimensional space. These solutions can be determined by means of suitable algorithms (cf. Armand and Malivert [5], Steuer [109]), but their composition over all possible $y$ (omitting dominated parts) is very difficult. A more viable technique consists in solving also the slave problem only *heuristically*, approximating the continuous Pareto front for fixed $y$ by a discrete (finite) number of points. [8]

**(ii) Stochastic problem** To obtain a bi-objective formulation for the stochastic problem described in Subsection 3.3.1 we focus on the special case where one objective function representing economic gains ($p = 1$) and one objective that addresses strategic gains ($q = 1$) is present. Furthermore we assume that the weights $v_r^{(1)}$ are normalized in such a way that the strategic objective function is comparable to the economic objective function. In this way the two objective function can be simply added instead of being combined by a weighted average to obtain our first objective function for the stochastic problem. The second objective function (3.21) addresses expected total overtime cost. Again considering the linear approximations of Subsection 3.2.7 the modified model can be formulated as follows:

[8] We also tested a *greedy heuristic* for solving the slave problem approximately. The solution quality of the obtained results, however, turned out as not too good. We leave an improvement of heuristics for approximately solving the slave problem as a topic of future research.

$$\text{(MPstoch) } \max \left( fc\,(y,x) = \left( f^{(1)}\,(y) + \bar{g}^{(1)}\,(x) \right), h\,(x) \right) \tag{3.31}$$

$$\text{s.t. } f^{(1)}\,(y) = \sum_{i=1}^{n} \mathbb{E}\left(w_i^{(1)}\right) y_i + \sum_{i<j} \mathbb{E}\left(w_{ij}^{(1)}\right) y_i y_j \tag{3.32}$$

$$\bar{g}^{(1)}(x) = \sum_{r=1}^{R} v_r^{(1)} \bar{\eta}_r \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \tag{3.33}$$

$$h\,(x) = -\mathbb{E}\left( \sum_{j=1}^{m} g_j \sum_{t=1}^{T} \left[ \sum_{k=1}^{K} \sum_{r=1}^{R} \sum_{i=1}^{n} c_{ik} U_i\, x_{krjt} - a_{jt} \right]^{+} \right) \tag{3.34}$$

$$x = \text{solution of the subproblem (SPstoch) to given } y \tag{3.35}$$

$$y_i \in \{0,1\} \qquad \forall i. \tag{3.36}$$

$$\text{(SPstoch) } \max \bar{g}^{(1)}\,(x) \tag{3.37}$$

$$\text{s.t. } \bar{g}^{(1)}(x) = \sum_{r=1}^{R} v_r^{(1)} \bar{\eta}_r \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \tag{3.38}$$

$$\text{(3.9) – (3.11), (3.17), (3.18)} \tag{3.39}$$

Let us discuss some properties of the modified stochastic model described above. For a *fixed* portfolio $y$, the work plan $x$ maximizing the first objective function $fc\,(x,y)$ is obtained by maximization of $\sum_{r=1}^{R} v_r^{(1)} \bar{\eta}_r \sum_{j=1}^{m} \varphi_r'(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs}$ over all $x = (x_{krjt})$ that are feasible in combination with the given $y$. Let us denote this work plan by $x^*(y)$. If we would have only a single feasible portfolio $y$, the solution $(y, x^*(y))$ would be guaranteed to be Pareto-optimal. Of course, in view of $h\,(x)$, it would (in general) not be the only Pareto-optimal solution corresponding to $y$, and if there exist also other feasible $y'$, it could even be that $(y, x^*(y))$ is dominated by some $(y', x')$. Nevertheless, the Pareto-optimal solutions amongst all solutions of the type $(y, x^*(y))$ are evidently good candidates for *approximating* the set of Pareto-optimal solutions of (3.20) – (3.22). Therefore, we modify our basic model by restricting ourselves to solutions of this type, imposing the additional constraint on $(y,x)$ that $x \in \arg\max_{x'} h(y,x')$.

The problem remains a bi-objective problem in this way, since the elements of the set

$\{(y, x^*(y)) \,|\, y \in \{0,1\}^n\}$ have to be evaluated with respect to both objective functions $fc\,(y,x)$ and $h\,(x)$ and Pareto-optima have to be determined, but the decision space is now reduced to the discrete finite set $\{0,1\}^n$. For the evaluation of each $y \in \{0,1\}^n$, an auxiliary problem, optimizing $x$ to the given $y$, has to be solved.

In this way, we obtain a similar hierarchical decomposition of the overall problem into a bi-objective discrete stochastic optimization *master problem* of determining Pareto-optimal portfolios $y$, and a single-objective continuous (and even linear) deterministic *subproblem* of determining the best work plan $x$ to given portfolio $y$.

### 3.4.1. General Approach

As the structure of the deterministic and stochastic problems share many properties the basic solution procedures are very similar. In the following paragraphs we describe the common features of the solution procedures for both cases.

Since the subproblem is an LP, its computational solution does not cause difficulties: it can be solved even for a very large number $KRmT$ of variables. Nevertheless, the subproblem has to be solved each time a solution $y$ of the master problem is to be evaluated. Therefore, it is important that the subproblem is solved as efficiently as possible. We use CPLEX version 11.0 for this purpose.

Contrary to the subproblem, the master problem belongs to a computationally hard class of problems. It is immediately seen that already the deterministic, single-objective special case contains the well-known *knapsack problem*, which is $\mathcal{NP}$-hard. The bi(multi)-objective situation and in the stochastic case the presence of uncertainty further increase the complexity. To obtain an approximate solution of the master problem we apply two multi-objective metaheuristics: the Nondominated Sorting Genetic Algorithm II (NSGA-II) by Deb et al.[33], and the Pareto Ant Colony (P-ACO) algorithm by Doerner et al.[35] (a brief description of the algorithms is given in Section 2.4). The solution $x$ returned by the procedure for the slave problem to the given portfolio $y$ is either unique or empty. If a non-empty solution $x = x(y)$ has been obtained for some $y$, the full vector of objective function values can be determined by the master procedure. Otherwise, the given portfolio $y$ does not admit a feasible work time array $x$. Thus, any multi-objective metaheuristic can be applied in the master procedure in a standard way, with the only exception that one has to take care of the case where to some $y$, no feasible $x$ is found.

In the stochastic case for most distributions of $U_i$, a direct evaluation of $h\,(x)$ by numerical methods is costly or even impossible: Determining the expected value of the expression $[\ldots]^+$ directly would require the computation of a convolution product of up to $n$ distributions. For this reason, we resort to Monte-Carlo simulation to obtain an estimate of $h(x)$ for each given $x$. To improve the variance of the estimate, we shall use the importance sampling technique (an introduction is given in Section 2.2).

Since we do not obtain exact evaluations of $h(x)$ in this way, but only stochastic estimates, there arises the question how the interplay between optimization and simulation should be handled in an efficient way. As the literature on the field called Simulation-Optimization shows (see, e.g., Pflug [98]), this is a highly nontrivial question. In our case, we are confronted with the additional complication that the optimization problem is *bi-objective*. Up to now, only few papers have addressed multi-objective discrete simulation-optimization problems and presented methods that could be used to tackle with such problems efficiently.

In this work, we apply a technique called *Adaptive Pareto Sampling* (APS) (cf. Subsection 2.4.1) developed in Gutjahr [54] in combination with the NSGA-II algorithm for the solution of stochastic multi-objective combinatorial optimization problems. A detailed discussion of convergence of the proposed approach and the considered application is given in Gutjahr and Reiter [55].

### 3.4.2. NSGA-II

In our application, where a solution consists of a binary vector $(y_1, \ldots, y_n)$, lends itself very well to the application of a genetic algorithm. Application dependent parts of the NSGA-II algorithm are implemented in a similar way as in the single objective case (see Gutjahr et al. [57]).

**(1) Encoding of a solution.** Each solution generated during the execution of the NSGA-II algorithm is encoded as a simple binary vector $(y_1, \ldots, y_n)$.

**(2) Generation of the initial population.** The initial population of chromosomes is generated with each chromosome $y$ consisting of $n$ bits that are chosen uniformly at random.

**(3) Crossover.** For crossover, we use a standard one-point crossover, which is applied to a fraction of the chromosomes of the population.

**(4) Mutation.** Mutation is implemented bit-wise by an independent random flip of each bit in each chromosome with a certain probability.

**(5) Constraint Handling.** In general crossover and mutation operations will generate solutions that may not be feasible, to cope with the situation when a solution is not feasible different approaches exist (for a brief introduction see 2.4.2). In the relevant literature on knapsack-type problems several repair mechanisms have been proposed to deal with the infeasibility of solutions. Greedy repair is reported to provide the best results (see Michalewicz and Arabas [89]). But in our application the complex constraints make it impossible to compute an analogue to the "weight"of an item in a knapsack problem. As greedy repair relies on benefit/weight ratios it is not applicable. To solve this problem we implemented a simpler repair mechanism, removing randomly selected projects from portfolio $y$ by setting the corresponding genes $y_i$ to zero, until feasibility is achieved.

**(6) Elite-preserving procedure.** As selection operator we use the standard elite-preserving procedure used by Deb et al. [33].

### 3.4.3. P-ACO

In this work we apply a variant of the P-ACO algorithm, where we use a pheromone update strategy of a MAX-MIN Ant System (Stützle et al. [113]).

**(1) Construction graph.** As in our application the search space is $S = \{0, 1\}$, each a solution consists of a binary vector $(y_1, \ldots, y_n)$, therefore we use a very simple construction graph, the so-called *chain* construction graph introduced in Gutjahr [53]. A single construction step corresponds to the assignment of a value 0 or 1 to one of the binary variables $y_i$. These values are assigned from left to right, i.e., for bit $1, 2, \ldots, n$.

**(2) Constraint handling.** In order to guarantee feasibility of the obtained solution, the following problem-dependent rule is used for our application: If up to now, the first $i - 1$ decisions variables have been set to the values $y_1, \ldots, y_{i-1}$, then the next variable $y_i$ is only allowed to be set to the value 1 if the project portfolio $(y_1, \ldots, y_i, 0, \ldots, 0)$ has a feasible work time array $x$. Whether this is the case or not is judged by the slave procedure. If it is the case, both values 0 and 1 are feasible for the current variable $y_i$; otherwise, only the value 0 is feasible.

**(3) Pheromone update.** In our experiments we use an iteration-best (round-best) pheromone update mechanism (see Dorigo and Stützle [36]). To avoid stagnation situations that can arise from the chosen pheromone update strategy, we use pheromone limits, as proposed by the MAX-MIN Ant System (Stützle et al. [113]).

**(4) Scalarization function** As shown in Section 2.4.3 P-ACO needs a scalarizing function. Different approaches (aggregation methods) can be used (see Section 2.1). The scalarization by weighted averages is a simple, intuitive approach to reduce multi-objective problems to single-objective ones; it assumes that the utility function of the decision maker is a linear function. In this work we use weighted Chebyshev distances which overcome the problems of the weighted averages approach. In a previous work Gutjahr et al.[57] we made experiments with both approaches, the achieved performance turned out as about the same for both choices in our experiments.

### 3.4.4. Importance Sampling

In our experiments, the random variables $U_i$ have been assumed as independent and modeled by triangular distributions $\Delta(\mathcal{B}_i, \mathcal{M}_i, \mathcal{W}_i)$, where $\mathcal{B}_i$, $\mathcal{M}_i$ and $\mathcal{W}_i$ are best case, most likely and worst case estimates ($\mathcal{B}_i < \mathcal{M}_i < \mathcal{W}_i$). To estimate objective function $h\,(x)$, a sample of $s$ scenarios $\omega_1, \ldots, \omega_s$ is drawn, where each scenario $\omega_\nu$ consists of a vector $U^{(\nu)} = (U_1^{(\nu)}, \ldots, U_n^{(\nu)})$ of i.i.d. random numbers $U_i^{(\nu)}$ distributed according to $\Delta(\mathcal{B}_i, \mathcal{M}_i, \mathcal{W}_i)$ $(i = 1, \ldots, n)$. According to (2.7), the estimator $\tilde{h}\,(x)$ for $h\,(x)$ is given by

$$\tilde{h}\,(x) = \frac{1}{s} \sum_{\nu=1}^{s} h\left(x, U^{(\nu)}\right) \tag{3.40}$$

where (cf. (3.34))

$$h(x, U^{(\nu)}) = -\sum_{j=1}^{m} g_j \sum_{t=1}^{T} \left[ \sum_{k=1}^{K} \sum_{r=1}^{R} \sum_{i=1}^{n} c_{ik}\, U_i^{(\nu)}\, x_{krjt} - a_{jt} \right]^+ . \tag{3.41}$$

To reduce the variance of the estimator $h\,(x)$ without paying the cost of increasing sample size, we use importance sampling (IS) in our experiments (see, e.g., Rubinstein and Kroese[107]). In our case, for estimating $h\,(x)$, we are only interested in events where the capacity $a_{jt}$ of some employee in some period is exceeded: if this is not the case, the term $\left[\sum_k \sum_r \sum_i c_{ik}\, U_i^{(\nu)}\, x_{krjt} - a_{jt}\right]^+$ in (3.41) is zero. This suggests to shift the distribution $\Delta(\mathcal{B}_i, \mathcal{M}_i, \mathcal{W}_i)$ of $U_i$ to $\Delta(\mathcal{B}_i, \mathcal{M}_i^+, \mathcal{W}_i)$ with some $\mathcal{M}_i^+$ satisfying $\mathcal{M}_i < \mathcal{M}_i^+ < \mathcal{W}_i$, such that the above-mentioned event occurs more frequently during sampling. The corresponding likelihood ratio is

$$\lambda(u;\, \mathcal{B}_i, \mathcal{M}_i, \mathcal{M}_i^+, \mathcal{W}_i) = \chi(u;\, \mathcal{B}_i, \mathcal{M}_i, \mathcal{W}_i) \,/\, \chi(u;\, \mathcal{B}_i, \mathcal{M}_i^+, \mathcal{W}_i),$$

where $\chi(u;\,\mathcal{B},\mathcal{M},\mathcal{W})$ denotes the probability density of the triangular distribution $\Delta(\mathcal{B},\mathcal{M},\mathcal{W})$ in point $u$. Note that the distributions $\Delta(\mathcal{B}_i,\mathcal{M}_i,\mathcal{W}_i)$ and $\Delta(\mathcal{B}_i,\mathcal{M}_i^+,\mathcal{W}_i)$ have the same support. By the assumed independence of the random variables $U_i$, we can multiply the likelihood ratios corresponding to the single variables $U_i$ to obtain the overall weight. Thus, we can replace (3.41) by

$$h^{IS}\left(x,U^{(\nu)}\right) = -\left(\sum_{j=1}^{m} g_j \sum_{t=1}^{T}\left[\sum_{k=1}^{K}\sum_{r=1}^{R}\sum_{i=1}^{n} c_{ik}\,U_i^{(\nu)}\,x_{krjt} - a_{jt}\right]^+\right) \left(\prod_{\ell=1}^{n}\lambda(U_\ell^{(\nu)};\,\mathcal{B}_\ell,\mathcal{M}_\ell,\mathcal{M}_\ell^+,\mathcal{W}_\ell)\right), \tag{3.42}$$

where $U_i^{(\nu)}$ is now sampled from $\Delta(\mathcal{B}_i,\mathcal{M}_i^+,\mathcal{W}_i)$ instead of $\Delta(\mathcal{B}_i,\mathcal{M}_i,\mathcal{W}_i)$ $(i=1,\ldots,n)$. To shift the distribution, a parameter $\alpha$ is used to determine $\bar{M}_i^+ = \bar{B}_i + \alpha(\bar{W}_i - \bar{B}_i)$ for each project $i$. We choose the parameter $\alpha$ as identical for each project.

Computational experiments confirmed that the amount of variance reductions is influenced by the parameter $\alpha$. In Figure 3.1, the results obtained by using different $\alpha$ values for a fixed work plan $x$ and fixed working capacities $a_{jt}$ are shown. Let $\sigma_{\tilde{h}}$ denote the standard deviation of the sample average estimate (3.40). As it can be seen from Figure 3.1, there is an optimal value $\alpha^*$ of $\alpha$ leading to the minimum standard deviation of the estimator. In the example of Figure 3.1, the optimal value of $\alpha$ is $\alpha^* \approx 0.6$.

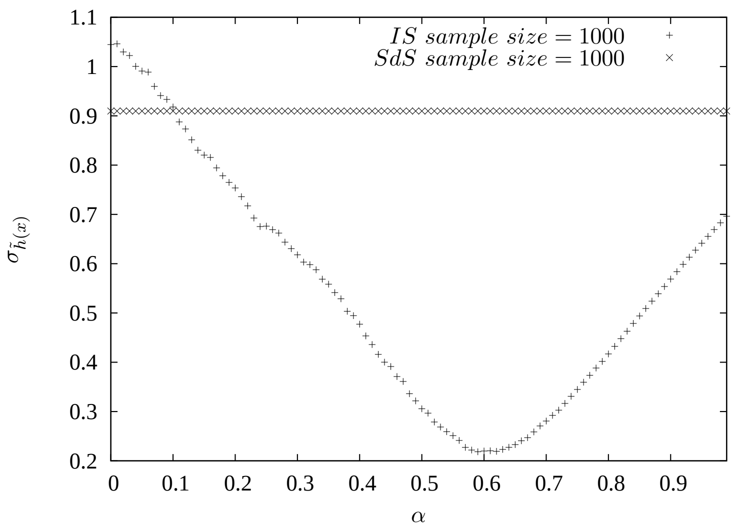



Figure 3.1.: Standard deviation of $\sigma_{\tilde{h}}$ for $s = 1000$ and different $\alpha$ values (SdS: standard sampling), IS: importance sampling).

The optimal value $\alpha^* = \alpha_i^*(y,x)$ of $\alpha$ for a given project $i$ depends in a rather complicated way on the parameters of the model, and there seems to be no chance to compute it in advance by means of some closed-form expression. Therefore, we tried to develop a heuristic rule for determining a suitable constant value $\alpha^c$ that can be used for $\alpha$ in *all* projects, effecting variance reduction in the case of each single project, although at different degrees. The intuition behind our heuristic rule is that the relation between work capacity and required work time plays a role for an appropriate choice of $\alpha$: If the available capacity is low compared to the required work time, then the event that the required work time exceeds capacity will be frequent during simulation, such that no importance sampling (or only a small probability shift) will be necessary. If, on the other hand, the available capacity is high compared to the required work time, then the event that the required work time exceeds capacity is a *rare* event, which makes a considerable probability shift for importance sampling advisable. In formal terms, we define the *actual working capacity*

$$A(y,x) = \sum_{j=1}^{m}\sum_{t=1}^{T} a_{jt} I\left(\sum_{k=1}^{K}\sum_{r=1}^{R} x_{krjt} > 0\right), \tag{3.43}$$

with $I$ denoting the indicator function, as the total capacity of all employees in all periods in which they actually do some work. To normalize the value of $A(y,x)$, we introduce the *relative* actual working capacity as $A^{rel}(y,x) = [A(y,x) - \bar{B}(y)] \,/\, [\bar{W}(y) - \bar{B}(y)]$, where $\bar{B}(y) = \sum_{i=1}^{n} \bar{B}_i y_i$ and $\bar{W}(y) = \sum_{i=1}^{n} \bar{W}_i y_i$. In the most interesting situation (but not necessarily always), in the best case, the actual working capacity $A(y,x)$ is sufficient to perform the projects of the portfolio $y$, such that $A(y,x) \geq \bar{B}(y)$, and in the worst case, it is *not* sufficient for that purpose, such that $A(y,x) \leq \bar{W}(y)$. In such a situation, $0 \leq A^{rel}(y,x) \leq 1$. By some pre-tests, we found that setting $\alpha^c(y,x) = 0.5 \cdot A^{rel}(y,x) + 0.5$ produces a (project-independent) $\alpha^c(y,x)$ that can be used as a good surrogate for the unknown, project-dependent optimal values $\alpha_i^*(y,x)$. This yielded variance reductions that were only by 7 – 12 % below those achieved by the optimal $\alpha_i^*$.

## 3.5. Test Instances

To evaluate the performance of the proposed methods we use two sets of test instances: randomly generated synthetic test cases of different size and type, as well as real-world instances provided by the E-Commerce Competence Center Austria (see Section 3.5.2). In the following section we describe the different test instances for the deterministic model as well as for the stochastic model.

### 3.5.1. Synthetic Test Cases

In the synthetically generated test cases[9] , we chose only one economic and one competence objective function, i.e., we set $p = 1$ and $q = 1$. We varied three factors that might influence the results of the used metaheuristics.

**(1) Instance size:** Since our standard real-world test instance consists of 18 candidate projects, we generated instances of this size also in the synthetic tests. For being able to give a comparison with exact solutions, we also studied a set of smaller instances. This gives the two instance classes: "small instances": 12 candidate projects, and "large instances": 18 candidate projects.

**(2) Tightness of capacity constraints:** We generated different *project sizes*, where the size $\zeta_i$ of project $i$ is defined as the overall effective work time required by project $i$. First, an average project size $\zeta$ was calculated as $\zeta = Tm\mu \,/\, n$, where $T$ is the number of periods, $m$ is the number of employees, and $n$ is the number of projects. The multiplier $\mu$ was used to define whether the capacity constraints are tight or loose: (i) tight: $\mu = 1.25$, (ii) loose: $\mu = 1.00$. Then for each project $i = 1, \ldots, n$, a random number $\xi_i$ was drawn from a uniform distribution on $[0, 1]$, and $\zeta_i$ was determined as $\zeta_i = (\zeta n \xi_i) \,/\, \left(\sum_{j=1}^{n} \xi_j\right)$. Finally, the project sizes $\zeta_i$ were split randomly into the effective work times $d_{kr}$ required by competency $r$ in task $k$ assigned to project $i$. In the synthetically generated test instances, we identified projects and tasks, i.e., we let each project consist of only one task. Economic benefits $w_i^{(1)}$ were determined as $\zeta_i \cdot (const + noise_i)$, where $noise_i$ is a random noise term with uniform distribution and mean zero.

**(3) Distribution of the competence weights:** We chose $R = 20$ competencies and investigated two distribution models: (i) Random: To each competence, a weight $v_r^{(k)}$ was assigned by drawing from a uniform distribution on $[0, 1]$. (ii) Counter-economic: In order to study the tradeoff between economic and competence benefits, we used the following rule to generate the competence weights. First, the projects were split into two groups

[9] In default of comparable integrated models, we cannot test our procedures on available benchmarks. There are test instances for special parts of our model. E.g., in Medaglia et al. [88], test cases for multi-objective project selection problems are obtained using Steuer's Steuer [110] ADBASE code. However, for our purposes, information on competence scores of employees, competence requirements of projects, learning rates etc. would have to be filled in, such that we found it more appropriate to generate our test cases from scratch.

with comparably low resp. high economic benefit. A competence weight $v_r^{(1)}$ drawn from a uniform distribution on $[0, 1]$ was assigned to competencies that were strongly required by the low-benefit group. The competence weights of other competencies were set to zero.

The possible combinations of levels for these three factors yield eight different types of test cases. For each type, ten independent test cases were generated randomly, leading us to obtain 80 test cases in total.

The other parameters were generated as follows:

- (a) Ready times and due dates: The values $\rho_k$ and $\delta_k$ for each task $k$ were determined by drawing two uniformly distributed random numbers $\xi^{(1)}$ and $\xi^{(2)}$ from $[0, 1]$, multiplying them by $T$, and rounding to integers. The smaller resulting number determines the ready time, the larger determines the due date.

- (b) Efficiencies: Initial efficiency values $\gamma_{jr1}$ for each employee $j$ and each competency $r$ in period 1 were drawn from a uniform distribution on $[0, 1]$. The initial competence scores $z_{jr1}$ were obtained by applying the inverse function $\varphi_r^{-1}$ to $\varphi_r$.

- (c) Capacities: The capacities $a_{jt}$ were determined based on a uniformly distributed random variable on $[0.5, 1]$.

- (d) Rates: Learning rates $\eta_r$ and deprecation rates $\beta_r$ for each competency $r$ were drawn from uniform distributions, using different scales with a factor of 800 between the maximum values of $\eta_r$ and $\beta_r$, respectively.

### 3.5.2. Real-World Test Cases

The Electronic Commerce Competence Center (EC3) Austria, a public-private partnership institution funded by the Austrian Federal Ministry of Economic Affairs and the City of Vienna as well as by twelve private enterprises was chosen as a real-world application case. The EC3 develops innovative technology within a cooperation network consisting of (i) the three major universities in Vienna (the University of Vienna, the Vienna University of Technology, and the Vienna University of Economics and Business Administration) and (ii) the twelve business partners. The EC3's goal is to support a fast transfer of knowledge between academic institutions and business partners. Covered research areas encompass such topics as methods of information access and information visualization, designs and mechanisms of Web-based systems, empirical business analysis by quantitative methods, and the evaluation of business ideas and models by market research. The following description of the data collection process is based on the article Gutjahr et al. [58].

According to the mathematical formulation of the optimization problem in Subsection 3.2.5, the EC3 was confronted with an extensive data requirement.A catalogue of relevant competencies as well as a competence scoring model had to be developed first. A draft for the catalogue included competencies from all four principal competence classes according to the categorization in Erpenbeck and Heyse[40], viz. personal, activity-oriented, social-communicative, and professional/methodological competence. Based on the results of an e-mail survey among EC3's researchers, this draft underwent several adaptations. Ultimately, only the professional and methodological competencies were kept. The final version consists of $R = 80$ competencies, these are structured into nine groups, viz. data analysis and business analytics, data organization including warehousing, digital economy and society, e-business components and foundations, functional business domains, symbol processing, digital technologies, economic activity categories, and methodological competence.

Objective and subjective evidences were used as competence indicators (cf. HR-XML Consortium[66]). The former distinguish formal qualifications in terms of certificates, diplomas, or publication records, and professional experience. The latter include the ratings of a researcher's competencies by him-/herself, by the scientific director, as well as by the peers in and by the head of the respective research group. The contribution of each objective evidence to each competence was specified based on background information such as curricula or journal citation indices. The subjective evidence was measured on a six-item ordinal scale according to the skill acquisition model in Dreyfus et al.[37]. Objective and subjective evidence was collected by surveying via e-mail $m = 28$ employees. , including the six heads of the research groups into which EC3 is structured, as well as the scientific director and six freelancers, in addition to the institution's 15 full- or part-time permanent researchers.

From the data gathered, the competence score $z_{jrt}$ was computed by taking the sum of the contributions of all objective evidence assigned to a researcher and adding an adjusted score obtained from the subjective competence ratings. Learning rates $\eta_r$ and knowledge depreciation rates $\beta_r$ were defined based on expert guesses. The knowledge depreciation rate was fixed at a rather optimistic level. Most competencies received the same learning and depreciation rates, except for several methodological competencies that were supposed to grow and diminish more slowly. The parameters $a$ and $b$ of the logistic function $\varphi_r(z)$ were chosen as identical for all competencies $r$, based on expert guesses.

For the majority of our test cases, a set of descriptions of $n = 18$ potential projects was used (data were extracted from projet plans). Three different measures of economic project benefit were provided, viz. the amount of third-party funding (ranging from approximately

200,000 Euro to no external funding at all), the overall rate of co-financing (covering the full interval from 0 to 100%), and the utility generated for business partners measured by means of an EC3-internal intellectual performance analysis. The projects included in the test data yield utility values up to 50.

Various settings of the competence weights $v_r^{(\kappa)}$ of the competencies were devised. Five competence profiles embarking on different strategies, e.g. the orientation towards technological projects, towards data analytic and empirical projects, or towards projects in the area of mobile business, were described. Moreover, several "opportunistic" competence profiles that primarily focus on those competencies that are required in many projects, an "indifferent" competence profile with equal relative importance assigned to all competencies, as well as an "ignorant" competence profile accounting only for those competencies not required in any project were designed.

### 3.5.3. Test Cases for the Stochastic Problem

As described in Paragraph 3.4.4, the random variables $U_i$ have been modeled by triangular distributions $\Delta(\mathcal{B}_i, \mathcal{M}_i, \mathcal{W}_i)$, where $\mathcal{B}_i$, $\mathcal{M}_i$ and $\mathcal{W}_i$ are best case, most likely and worst case estimates ($\mathcal{B}_i < \mathcal{M}_i < \mathcal{W}_i$).

To obtain these values, we start by estimating the best case work time $\bar{B}_i$, the most likely work time $\bar{M}_i$ and the worst case work time $\bar{W}_i$ for each project $i$. The expected value of a $\Delta(\bar{B}_i, \bar{M}_i, \bar{W}_i)$ distribution is given as $(\bar{B}_i + \bar{M}_i + \bar{W}_i)/3$, which has to be equal to the expected required effective work time $\sum_k \sum_r c_{ik} d_{kr}$ of project $i$. Therefore, we set $d_i^{total} = (\bar{B}_i + \bar{M}_i + \bar{W}_i)/3$ and partition this estimated effort into effort estimates $d_k$ for the tasks $k$ assigned to project $i$, and after that, we further partition each estimate $d_k$ into effort estimates $d_{kr}$ for the work packages assigned to task $k$. The parameters of the distribution of $U_i$ result then as $\mathcal{B}_i = \bar{B}_i / d_i^{total}$, $\mathcal{M}_i = \bar{M}_i / d_i^{total}$, $\mathcal{W}_i = \bar{W}_i / d_i^{total}$, which gives $\mathbb{E}(U_i) = (\mathcal{B}_i + \mathcal{M}_i + \mathcal{W}_i)/3 = 1$ as required.

To evaluate the performance of the proposed methods ten different test instances (derived from the real-world instances) are used. The deterministic parameters of the test instances were derived from a real-world application case (E-Commerce Competence Center Austria). This application as well as the way the necessary parameters have been obtained is already described in Subsection 3.5.2, so we do not give details here, but focus on the additional parameter choices required for the stochastic extension of the model.

We used test instances with $n = 12$ candidate projects, $m = 20$ employees, $R = 20$ competencies and a planning horizon of $T = 24$ periods. Projects consist of 1 to 3 tasks. As described in Subsection 3.5.1, we varied (among others) two main factors that may influence the results of the used metaheuristics: (i) the tightness of the capacity constraints, and (ii) the joint distribution of the expected economic gains $\mathbb{E}(w_i^{(1)})$ and the strategic gains $v_r^{(1)}$. Synergy and cannibalization effects did not play a role in the described application, so $\mathbb{E}(w_{is}^{(1)})$ was chosen as zero for all $i$, $s$. For each of the two factors, two levels were defined: tight and loose capacity constraints; no correlation resp. negative correlation between economic and strategic gains.

Two different groups of test instances are considered. In five test instances, the distributions of the random variables $U_i$ are the same for each project $i$. The other five test instances consider varying distributions of the random variables $U_i$, this may be used to model projects that are more risky than others. The possible combinations of levels for factors (i) and (ii) yield four different types of test instances. In each group, one test instance of each type is included, plus one extra test instance of the (most interesting) type "tight capacity constraints" and "negatively correlated gains". The parameters for the distribution of the random variable $U_i$ for each project $i$ were determined as follows: For $\bar{W}_i$, a uniformly distributed random number from $\bar{w}_i \in [1.5, 2.0]$ was drawn, $\bar{W}_i$ was then calculated as $\bar{W}_i = \bar{w}_i d_i^{total}$. For $\bar{M}_i$, again a random number $\bar{m}_i \in [0.5, 1.0]$ was drawn to calculate $\bar{M}_i = (3 - \bar{w}_i)\bar{m}_i d_i^{total}$. Finally, $\bar{B}_i$ was calculated as follows: $\bar{B}_i = (3 - \bar{w}_i)(1 - \bar{m}_i)d_i^{total}$. In this way, $d_i^{total} = (\bar{B}_i + \bar{M}_i + \bar{W}_i)/3$ is always satisfied, as required. For the test instances that use the same distribution for each project $i$, only one set of parameters was created, which was then used for each project.

In Table 3.7 in Subsection 3.6.3 the first two columns give a survey on the resulting test instances. The second column encodes the instance type according to the following scheme: The first character represents the distribution (e equal, v varying), the second character the constraints (t tight, l loose), and the third character the gains (c correlated, u uncorrelated).

## 3.6. Results

### 3.6.1. Results for Synthetic Test Cases

For each of the 80 test cases, we performed ten runs of our metaheuristics with different seeds. For the test cases of small instance size ($n = 12$), we compared the results of the metaheuristics with a procedure where on the master problem level, the multi-objective metaheuristic was replaced by complete enumeration (CE) and determination of the exact Pareto front. These runs required about 4.1 hours per test case. We gave the metaheuristics 5 % of the runtime of the CE runs, i.e., each metaheuristic was given 12 minutes computation time. Some preliminary experiments with different instance sizes showed that by allowing the runtime for the metaheuristics to increase quadratically in $n$, the algorithms scaled reasonably, such that we decided to execute the metaheuristics on the test cases with large instance size ($n = 18$) using $12 \cdot (18/12)^2 = 27$ minutes runtime.

For the evaluation of the quality of the solution sets delivered by the multi-objective metaheuristics, we chose three measures: the hypervolume measure, the spacing measure and the coverage measure (a detailed description of the measures is given in Section 2.3).

Tables 3.1 and 3.2 show the experimental results for the 80 test instances. For the comparison between P-ACO and NSGA-II with respect to hypervolume and spacing measure, a two-sided Mann-Whitney test was used to judge statistical significance of superiority results. Significantly superior entries were marked by stars. For the coverage and spacing measures, no significance tests were performed because the condition of independent basic variables is not satisfied for the indicated aggregations. In the case of the small test instances (Table 3.1), we also present the hypervolumes of the exact solutions, obtained by solving the master problem by CE.

From Table 3.1, it can be seen that for the small test instances, P-ACO and NSGA-II are comparable in their performance with respect to hypervolume and spacing measure, but NSGA-II is better than P-ACO with respect to the coverage measure in 35 of 40 cases. For the large synthetic test instances, Table 3.2 shows that NSGA-II clearly outperforms P-ACO. For the small instances where exact solutions are known, we see that in most cases, the hypervolume values achieved by the metaheuristics deviate from those of the exact solutions by less than 10 percent.

| t.c. | Exact | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|---|
| | $I_H$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ |
| 1 | 0.9339 | 0.8578 | 0.0524 | 0.1703 | 0.8394 | 0.0505 | 0.5409 |
| 2 | 0.9502 | 0.8981 | 0.0653 | 0.2856 | 0.9044 | 0.0708 | 0.5085 |
| 3 | 0.9293 | 0.8509 | 0.0576 | 0.1746 | 0.8720 | 0.0470 | 0.4752 |
| 4 | 0.9287 | 0.8513 | 0.0740 | 0.2225 | 0.8687 | 0.0652 | 0.4850 |
| 5 | 0.9497 | 0.8829 | 0.0611 | 0.2402 | 0.8901 | 0.0493 | 0.5320 |
| 6 | 0.9546 | 0.8560 | 0.0493 | 0.1291 | 0.9057** | 0.0518 | 0.7828 |
| 7 | 0.9149 | 0.8372 | 0.0804 | 0.3083 | 0.8426 | 0.0921 | 0.4835 |
| 8 | 0.9214 | 0.8410 | 0.0595 | 0.2514 | 0.8379 | 0.0462 | 0.5060 |
| 9 | 0.8979 | 0.8217* | 0.0515 | 0.2442 | 0.7679 | 0.0683 | 0.3888 |
| 10 | 0.9116 | 0.8132 | 0.0631 | 0.1688 | 0.8357 | 0.0566 | 0.5884 |
| 11 | 0.9429 | 0.8404 | 0.0460** | 0.2433 | 0.8230 | 0.0816 | 0.4794 |
| 12 | 0.9391 | 0.8956** | 0.0603 | 0.2987 | 0.8462 | 0.1073 | 0.2238 |
| 13 | 0.9220 | 0.8629** | 0.0602 | 0.3609 | 0.8172 | 0.0871 | 0.2882 |
| 14 | 0.9680 | 0.8785 | 0.0541 | 0.1746 | 0.8783 | 0.0668 | 0.5643 |
| 15 | 0.9470 | 0.8710** | 0.0625 | 0.1707 | 0.8080 | 0.0624 | 0.3638 |
| 16 | 0.9580 | 0.8659 | 0.0541 | 0.2355 | 0.8402 | 0.0723 | 0.3449 |
| 17 | 0.9335 | 0.8523* | 0.0542 | 0.2478 | 0.8224 | 0.0821 | 0.3389 |
| 18 | 0.9355 | 0.8337 | 0.0616 | 0.1867 | 0.8342 | 0.0835 | 0.3912 |
| 19 | 0.9366 | 0.8510* | 0.0572 | 0.2819 | 0.8236 | 0.0475 | 0.4091 |
| 20 | 0.9585 | 0.8696 | 0.0598 | 0.2649 | 0.8668 | 0.0785 | 0.4500 |
| 21 | 0.9652 | 0.8995 | 0.0532 | 0.3701 | 0.8927 | 0.0510 | 0.4287 |
| 22 | 0.9245 | 0.8219 | 0.0721 | 0.1652 | 0.8528** | 0.0506* | 0.6234 |
| 23 | 0.9370 | 0.8403 | 0.0717 | 0.2039 | 0.8676 | 0.0608 | 0.5854 |
| 24 | 0.9396 | 0.8735* | 0.0457 | 0.6572 | 0.8354 | 0.0613 | 0.2380 |
| 25 | 0.9411 | 0.8745 | 0.0634 | 0.3852 | 0.8710 | 0.0581 | 0.3733 |
| 26 | 0.9261 | 0.8442 | 0.0585 | 0.2714 | 0.8537 | 0.0351* | 0.4750 |
| 27 | 0.9386 | 0.8454 | 0.0455 | 0.3471 | 0.8418 | 0.0439 | 0.5374 |
| 28 | 0.9390 | 0.8440 | 0.0557 | 0.2893 | 0.8625 | 0.0630 | 0.4702 |
| 29 | 0.9166 | 0.8671 | 0.0543 | 0.3048 | 0.8556 | 0.0568 | 0.3831 |
| 30 | 0.9447 | 0.8611 | 0.0780 | 0.3175 | 0.8720 | 0.0481* | 0.4923 |
| 31 | 0.9348 | 0.8437 | 0.0622 | 0.2624 | 0.8709** | 0.0642 | 0.6225 |
| 32 | 0.9548 | 0.8771 | 0.0404 | 0.2565 | 0.8953 | 0.0346 | 0.5069 |
| 33 | 0.9430 | 0.8477 | 0.0575 | 0.1708 | 0.8889** | 0.0435 | 0.6402 |
| 34 | 0.9269 | 0.8666 | 0.0583 | 0.2137 | 0.8755 | 0.0432 | 0.5373 |
| 35 | 0.9308 | 0.8699 | 0.0583 | 0.3067 | 0.8783 | 0.0563 | 0.4568 |
| 36 | 0.9054 | 0.8042 | 0.0615 | 0.1077 | 0.8501** | 0.0445 | 0.7046 |
| 37 | 0.9126 | 0.8805** | 0.0501 | 0.4391 | 0.8575 | 0.0471 | 0.3686 |
| 38 | 0.9176 | 0.8809 | 0.0442* | 0.4706 | 0.8515* | 0.0685 | 0.2968 |
| 39 | 0.9489 | 0.8618 | 0.0521 | 0.2844 | 0.8793 | 0.0459 | 0.5050 |
| 40 | 0.9299 | 0.8366 | 0.0712 | 0.3102 | 0.8519 | 0.0626 | 0.4781 |

Table 3.1.: Mean values of the hypervolume, the spacing measure and the coverage measure over 10 runs for the small synthetic test cases. Stars * resp. ** indicate statistically significant superiority at level $\alpha = 0.05$ resp. 0.01.

| t.c. | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|
| | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ |
| 1 | 0.8548 | 0.0869 | 0.1425 | 0.8634 | 0.0466 | 0.5719 |
| 2 | 0.8348 | 0.0563 | 0.1132 | 0.8569* | 0.0616 | 0.6450 |
| 3 | 0.7799 | 0.0598 | 0.0838 | 0.8280** | 0.0639 | 0.6543 |
| 4 | 0.8181 | 0.0657 | 0.0963 | 0.8204 | 0.0484 | 0.5563 |
| 5 | 0.8306 | 0.0678 | 0.0943 | 0.8868** | 0.0660 | 0.6417 |
| 6 | 0.8347 | 0.0486 | 0.0747 | 0.9082** | 0.0504 | 0.8260 |
| 7 | 0.8250 | 0.0668 | 0.0334 | 0.8750** | 0.0630 | 0.8172 |
| 8 | 0.8414 | 0.0575 | 0.1064 | 0.8641** | 0.0496 | 0.6022 |
| 9 | 0.7850 | 0.0700 | 0.0802 | 0.8395** | 0.0629 | 0.7313 |
| 10 | 0.8298 | 0.1006 | 0.0384 | 0.9228** | 0.0469* | 0.8317 |
| 11 | 0.8501 | 0.0430 | 0.1342 | 0.8680 | 0.0390 | 0.4926 |
| 12 | 0.8126 | 0.0786 | 0.0964 | 0.8730** | 0.0468* | 0.6471 |
| 13 | 0.8245 | 0.0721 | 0.1366 | 0.8464 | 0.0515 | 0.5562 |
| 14 | 0.8068 | 0.0457 | 0.0514 | 0.8656** | 0.0402 | 0.6953 |
| 15 | 0.8543 | 0.0564 | 0.0900 | 0.8952** | 0.0351 | 0.5802 |
| 16 | 0.8809 | 0.0530 | 0.1253 | 0.8795 | 0.0359 | 0.5275 |
| 17 | 0.8592 | 0.0547 | 0.1091 | 0.8920** | 0.0432 | 0.5880 |
| 18 | 0.8474 | 0.0473 | 0.1211 | 0.8649* | 0.0511 | 0.5350 |
| 19 | 0.8625 | 0.0410 | 0.0858 | 0.8856* | 0.0554 | 0.6454 |
| 20 | 0.8260 | 0.0566 | 0.0616 | 0.8662** | 0.0390 | 0.6987 |
| 21 | 0.8305 | 0.0731 | 0.1626 | 0.8522 | 0.0710 | 0.5152 |
| 22 | 0.7786 | 0.0451 | 0.0908 | 0.7834 | 0.0615 | 0.6260 |
| 23 | 0.8185 | 0.0823 | 0.1156 | 0.8698** | 0.0476* | 0.7280 |
| 24 | 0.7662 | 0.0766 | 0.0545 | 0.8381** | 0.0511 | 0.7566 |
| 25 | 0.7804 | 0.0651 | 0.1209 | 0.8728** | 0.0492 | 0.7260 |
| 26 | 0.8239 | 0.0553 | 0.1670 | 0.8341 | 0.0470 | 0.6088 |
| 27 | 0.8189 | 0.0452 | 0.2303 | 0.8337 | 0.0364 | 0.5622 |
| 28 | 0.7704 | 0.0496 | 0.1032 | 0.8136** | 0.0679 | 0.6922 |
| 29 | 0.8207 | 0.0674 | 0.1725 | 0.8825** | 0.0525 | 0.7262 |
| 30 | 0.7939 | 0.0720 | 0.1315 | 0.8363 | 0.0571 | 0.6962 |
| 31 | 0.8557 | 0.0453 | 0.1590 | 0.8777** | 0.0398 | 0.5388 |
| 32 | 0.8078 | 0.0485 | 0.0983 | 0.8645** | 0.0469 | 0.7617 |
| 33 | 0.8384 | 0.0514 | 0.0741 | 0.8735** | 0.0425 | 0.6560 |
| 34 | 0.8797 | 0.0432 | 0.2601 | 0.8774 | 0.0304* | 0.5222 |
| 35 | 0.8393 | 0.0517 | 0.1211 | 0.8543 | 0.0404 | 0.5854 |
| 36 | 0.8099 | 0.0635 | 0.1368 | 0.8499** | 0.0426* | 0.6332 |
| 37 | 0.7917 | 0.0665 | 0.1108 | 0.8704** | 0.0330* | 0.7114 |
| 38 | 0.8237 | 0.0645 | 0.0793 | 0.8744** | 0.0561 | 0.7243 |
| 39 | 0.7934 | 0.0475 | 0.1155 | 0.8371** | 0.0510 | 0.6429 |
| 40 | 0.7962 | 0.0579 | 0.0779 | 0.8479** | 0.0423 | 0.7024 |

Table 3.2.: Mean values of the hypervolume, the the spacing measure and the coverage measure over 10 runs for the large synthetic test cases. Stars as in Table 3.1.

An interesting question is whether there is any pattern of the hypervolume lost between the solutions proposed by the metaheuristics and the exact Pareto front. A good way to judge this visually is to look at a plot of the *attainment function* (see Section 2.3.1) because it comprises several test runs. In Figure 3.2, this is illustrated for a selected (small) test instance. The 50 % attainment function of each of the two heuristic algorithms (i.e., the border of the area of points in the plane dominated by the proposed solution sets of at least 50 % of the runs) is compared to the attainment function of the exact Pareto front. Obviously, the extreme solution optimizing objective 2 is approximated a *little* bit better than the extreme solution optimizing objective 1. We observed the same trend also for the other test instances.

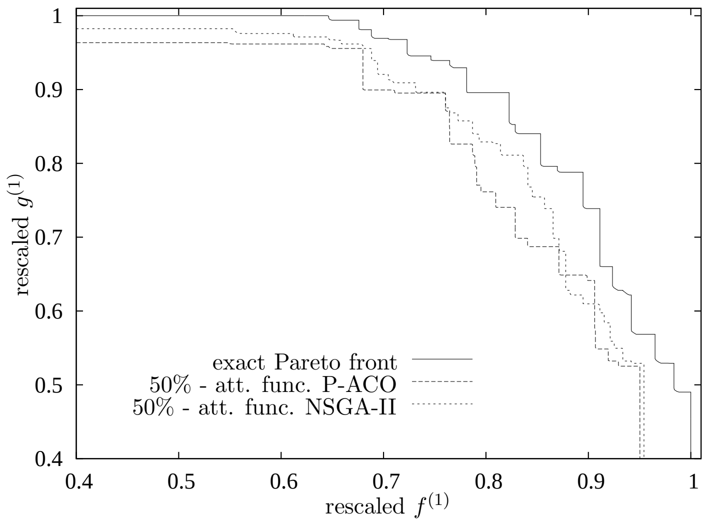
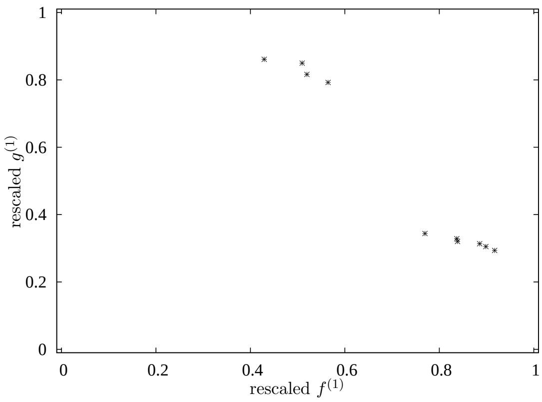



Figure 3.2.: Pareto front and 50 % attainment functions of the P-ACO and the NSGA-II solutions for a selected synthetic test instance.

We were interested in analyzing how well the LP solution approach performs compared to the solution of the slave problem by a heuristic. For this purpose, the algorithms were run again on the set of small artificial instances, with the LP solver for the solution of the slave problem replaced by a simple greedy heuristic (called SchedSA) for scheduling and staff assignment, developed in Gutjahr et al. [57] for a single-objective version of our model and described there in detail.

The results showed that in the case of two objectives, the greedy approach performed not too well; SchedSA sometimes produced hypervolumes lying by 40 % or more below that of the exact Pareto front. Figure 3.3 illustrates the performance of the greedy heuristic for a run of a special test instance. In the left picture, the image points in objective space of the portfolios proposed by SchedSA are shown. The right picture shows the true Pareto front (squares). Moreover, for the portfolios proposed by SchedSA, we also computed those objective function values they could achieve if the slave problem was solved optimally; this increases the value of the competence-related objective function 2. The resulting points are the crosses in the right picture.[10] Even then, however, we see that the points achieved by SchedSA are still rather far from the Pareto front.



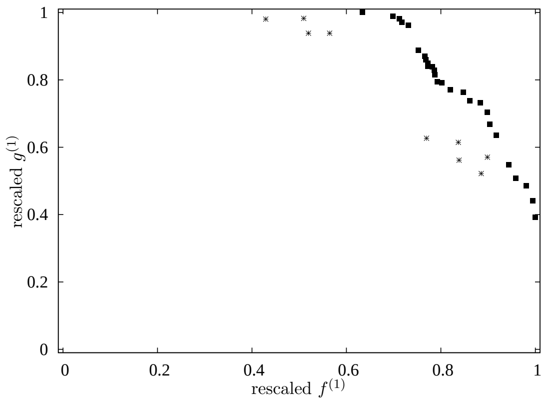




Figure 3.3.: Left picture: solution (in objective space) proposed by SchedSA. Right picture: exact Pareto front (filled squares), and solution (in objective space) corresponding to the portfolios proposed by using the greedy heuristic, evaluated with optimally solved sub-problem (crosses).

To compare the performance of the two algorithms in the case of more than two objectives, additional economic objective functions have been added to the first large synthetical test case. Note that in this first test case (with only two objectives), NSGA-II has turned out as slightly superior (cf. Table 3.2). As seen from Table 3.3, with growing number of objectives, P-ACO becomes superior with respect to hypervolume. Figure 3.4, where the case of two objectives from Table 3.2 has been added as the leftmost pair of points, visualizes this trend. With respect to the spacing measure, on the other hand, NSGA-II preserves its superiority also for a higher number of objective functions.

[10]The rationale behind this analysis is that after a portfolio has been selected with the help of the decision support system, an experienced manager could fine-tune both schedule and staff assignment for this portfolio, solving in this way the sub-problem almost optimally.

| nb. objectives | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|
| | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ |
| 2 | 0.8548 | 0.0869 | 0.1425 | 0.8634 | 0.0466 | 0.5719 |
| 3 | 0.8239** | 0.0973 | 0.2191 | 0.7106 | 0.1263 | 0.1627 |
| 4 | 0.6753* | 0.0949 | 0.1936 | 0.6087 | 0.1291 | 0.0867 |
| 5 | 0.5771** | 0.1110 | 0.2565 | 0.5001 | 0.1489* | 0.1413 |
| 6 | 0.5654** | 0.1412 | 0.2499 | 0.4092 | 0.1551 | 0.0843 |
| 7 | 0.4941** | 0.1476 | 0.2205 | 0.3682 | 0.1578 | 0.1030 |
| 8 | 0.4848** | 0.1377 | 0.2334 | 0.3249 | 0.1614 | 0.0602 |

Table 3.3.: Mean values of the hypervolume, the spacing measure and the coverage measure over 10 runs for the first large synthetical test case with an increasing number of economic objectives ($q = 1,\ p = 1 \dots 7$). Stars as in Table 3.1.

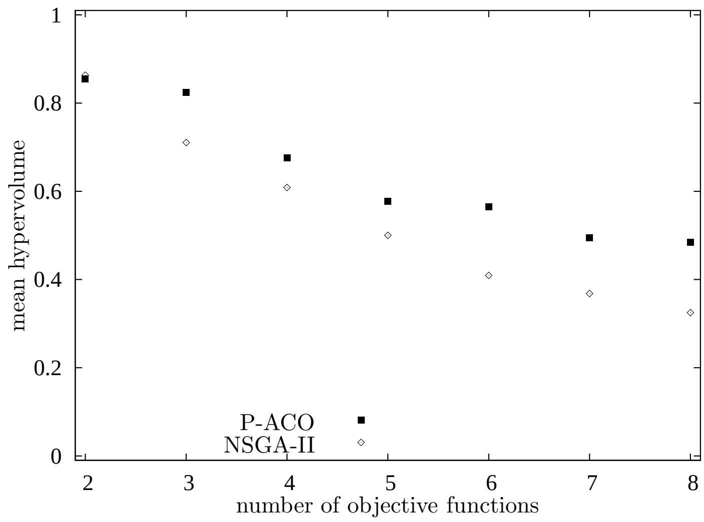



Figure 3.4.: Comparison of the mean hypervolumes over 10 runs for the large synthetical test case with an increasing number of economic objectives.

For dealing with the case $q > 1$, also the slave problem was solved heuristically, as outlined before. The results for ten random test instances with $p = 1$ and $q = 2$ are given below. As one can see, the GA-based approach turns out as superior. However, the reservations indicated before concerning the reduced solution quality in the case of a heuristic solution of the subproblem have to be kept in mind.

| t.c. | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|
| | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ |
| 1 | 0.7419 | 0.0270 | 0.0092 | 0.9621** | 0.0381 | 0.9143 |
| 2 | 0.8246 | 0.0435 | 0.0760 | 0.8786* | 0.0559 | 0.5151 |
| 3 | 0.7792 | 0.0439 | 0.1149 | 0.8004 | 0.0503 | 0.6426 |
| 4 | 0.7610 | 0.0545 | 0.0157 | 0.9278** | 0.0489 | 0.9064 |
| 5 | 0.6597 | 0.0833 | 0.0701 | 0.7773** | 0.0692 | 0.6929 |
| 6 | 0.8187 | 0.0136 | 0.0295 | 0.9956** | 0.0351* | 0.9867 |
| 7 | 0.8037 | 0.0560* | 0.0365 | 0.8002 | 0.0392 | 0.5008 |
| 8 | 0.8095 | 0.0570 | 0.1198 | 0.8293 | 0.0464 | 0.4862 |
| 9 | 0.6909 | 0.0543 | 0.0771 | 0.8096** | 0.0575 | 0.5157 |
| 10 | 0.7055 | 0.0760 | 0.0812 | 0.8447** | 0.0536 | 0.5959 |

Table 3.4.: Mean values of the hypervolume, the spacing measure and the coverage measure over 10 runs for 10 random test instances with $p = 1$ and $q = 2$. Stars as in Table 3.1.

### 3.6.2. Results for the Real-World Application

| t.c. | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|
| | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ | $\mu(I_H)$ | $\mu(I_{SP})$ | $\mu(I_{CO})$ |
| 1 | 0.6182* | 0.0511 | 0.5323 | 0.5693 | 0.0459 | 0.1707 |
| 2 | 0.8323** | 0.0964 | 0.7901 | 0.6011 | 0.1192 | 0.0993 |
| 3 | 0.8681* | 0.0813 | 0.7153 | 0.6834 | 0.1033 | 0.1277 |
| 4 | 0.9076** | 0.1108 | 0.8604 | 0.6922 | 0.1555 | 0.0679 |
| 5 | 0.8872** | 0.1543 | 0.5971 | 0.7343 | 0.0946 | 0.0710 |
| 6 | 0.8229** | 0.0550 | 0.7011 | 0.6503 | 0.1142 | 0.1865 |
| 7 | 0.9068** | 0.1164* | 0.8946 | 0.6242 | 0.1788 | 0.0294 |
| 8 | 0.7172 | 0.1692 | 0.2700 | 0.8162* | 0.1075 | 0.5945 |
| 9 | 0.5599 | 0.0787 | 0.5568 | 0.5050 | 0.0711 | 0.1950 |
| 10 | 0.6031 | 0.0444 | 0.4705 | 0.5794 | 0.0518 | 0.2361 |

Table 3.5.: Mean values of the hypervolume, the spacing measure and the coverage measure over 10 runs for the real-world test cases. Stars as in Table 3.1.

Table 3.5 provides the results of the comparison between P-ACO and NSGA-II. Surprisingly, we can observe that P-ACO seems to outperform NSGA-II now. A first intuitive explanation for this phenomenon is that although we increased the runtime for the real-world test cases from 27 to 120 minutes, compared to the large synthetic cases, this seems not to have compensated for the increment of the number of competencies from 20 to 80 and of the number of objectives from 2 to 3. As a consequence, the given runtime in the real-instance case may still be too low for NSGA-II to deploy its full strengths. (Note that P-ACO can partially compensate for low computation time by its more "greedy" constraint handling mechanism. For larger runtime, this advantage turns into a disadvantage.) To test this hypothesis, we further increased the runtime for the real-world instances from 2 to 4 hours. In this case, P-ACO and NSGA-II performed almost equally well, perhaps even with a slight advantage for NSGA-II.

The following table contains the results for the real-world test cases, if, compared to Table 3.5, the runtime for each of the algorithms is increased from 2 to 4 hours per run of a test case.

| t.c. | P-ACO | | | NSGA-II | | |
|---|---|---|---|---|---|---|
| | $\mu\,(I_H)$ | $\mu\,(I_{SP})$ | $\mu\,(I_{CO})$ | $\mu\,(I_H)$ | $\mu\,(I_{SP})$ | $\mu\,(I_{CO})$ |
| 1 | 0.8156* | 0.0215 | 0.7000 | 0.6665 | 0.0527 | 0.2000 |
| 2 | 0.8226 | 0.0528 | 0.0000 | 0.9353** | 0.0610 | 0.7850 |
| 3 | 0.8177 | 0.0440 | 0.0601 | 0.9669** | 0.0634 | 0.3917 |
| 4 | 0.8351 | 0.0207 | 0.0200 | 0.9789** | 0.0420 | 0.5917 |
| 5 | 0.8519 | 0.0296 | 0.0000 | 0.9865** | 0.0329 | 0.5067 |
| 6 | 0.8565 | 0.0553 | 0.0702 | 0.9515** | 0.0726 | 0.4283 |
| 7 | 0.8594 | 0.0910 | 0.0933 | 0.9559** | 0.0379* | 0.6150 |
| 8 | 0.8239 | 0.0009 | 0.4750 | 0.7917 | 0.0076 | 0.3833 |
| 9 | 0.8942** | 0.0411 | 0.6933 | 0.7033 | 0.0634 | 0.1350 |
| 10 | 0.7860** | 0.0204 | 0.4833 | 0.6831 | 0.0427 | 0.1200 |

Table 3.6.: Mean values of the hypervolume, the spacing measure and the coverage measure over 10 runs for the real-world test cases with 4 hours computation time per run. Stars as in Table 3.1.

A second factor possibly disadvantaging NSGA-II in the (three-objective) real-world test cases may be the property observed in Wagner et al.[123] that NSGA-II is especially strong in the bi-objective case. In order to study the influence of the number of objectives on the solution quality, we took the first large synthetic test case (cf. Table 3.2) and successively increased the number objectives from 2 to 8 (cf. Table 3.2, Figure 3.4). Indeed, it turned out that whereas NSGA-II dominated P-ACO for two objectives, P-ACO was significantly better than NSGA-II for the cases of three to eight objectives, with a slightly increasing gap.

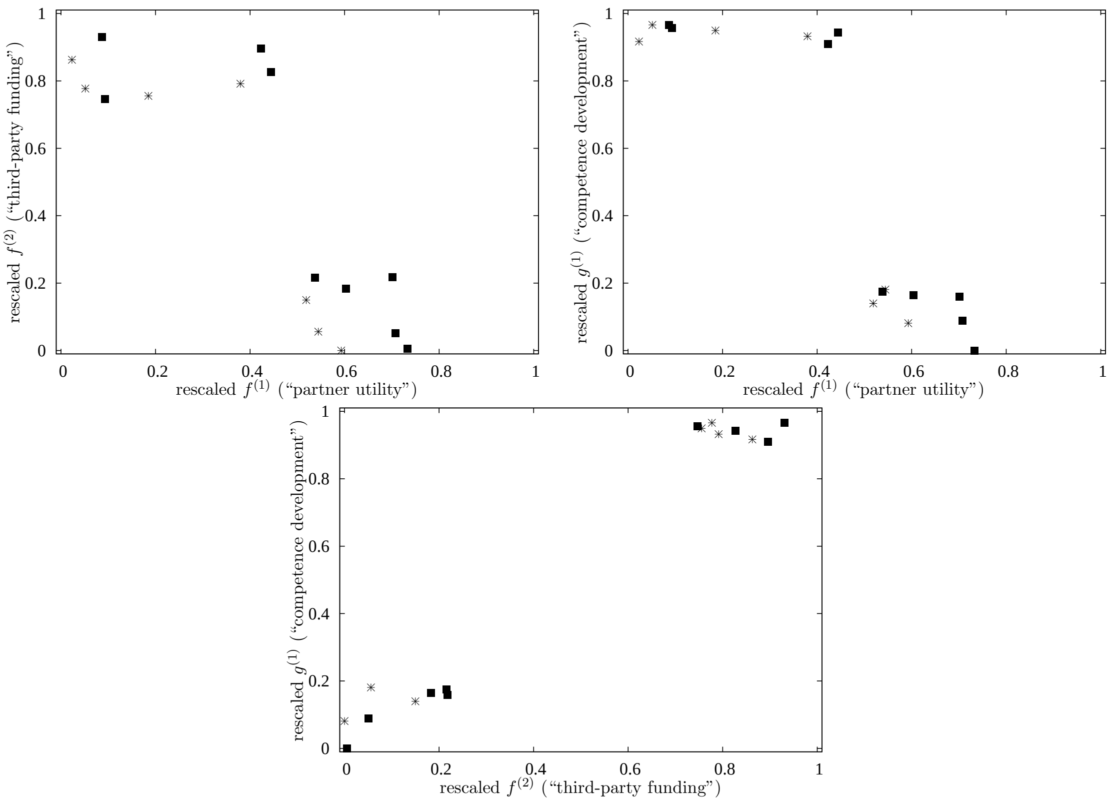



Figure 3.5.: Proposed Pareto-optimal solutions for real-world test case 1, projected in objective space to the three planes given by two of the axes. Filled squares: P-ACO. Crosses: NSGA-II.

For test case 1, we show the solutions proposed by P-ACO and NSGA-II from different views: The plots in Figure 3.5 present projections of the proposed Pareto-optimal solutions in objective space to the planes defined by the three possible pairs of objective functions. Objective function 1 gives partner utility, objective function 2 represents third-party funding, and objective function 3 indicates the competence gain. The objective function values have been normalized by mapping the obtained range of values to the interval $[0, 1]$.

One can see that on the (proposed) Pareto front, objective functions 1 and 2 are negatively correlated (higher partner utility is associated with lower third-party funding), the same holds for objective functions 1 and 3 (higher partner utility is associated with lower competence gain). Objective functions 2 and 3 are positively correlated (higher third-party is associated with higher competence gain).

Finally, in order to test whether our approach is still computationally feasible when the number of projects is increased, we extended the standard test instances just described by 22 additional candidate project descriptions. This produced a test case with a total number of 40 projects. P-ACO and NSGA-II performed almost equally well for this test instance, with a hypervolume of 0.2503 for P-ACO and of 0.2394 for NSGA-II.

### 3.6.3. Results for the Stochastic Problem

For our instances, complete enumeration combined with simulation using a sample size of $10^4$ can be performed within reasonable time to get a reference set approximating the set of Pareto-optimal solutions sufficiently well.

The following parameters of the APS algorithm were used for the tests: (i) The number of iterations for the APS algorithm was set to 100. (ii) A fixed sample size for solution proposal $s_1 = 100$ was used. (iii) The sample size for solution evaluation was increased according to the scheme $\bar{s}_\kappa = 1000 + 90 \cdot \kappa$ ($\kappa = 1, \ldots, 100$). Thus, in the last iteration, the sample size was $10^4$.

For each of the ten test cases, we performed ten independent runs of our metaheuristic. We compared the results with a reference set that was derived by complete enumeration (CE) on the level of the master problem with sample size $10^4$. These runs required about three hours per test instance. Each run of our heuristic was given 100 sec, which is about 1 % of the runtime of the CE runs. Table 3.7 shows the experimental results for the 10 test instances. In the third column of the table, the hypervolume value of the reference set $B$ obtained by CE is shown.[11] We see that in nine of the ten test instance, APS yields very good solutions, although spending only 1 % of the runtime of CE. For test instance 9, APS achieves (in the average) a hypervolume which is about 1.6 % below that of the reference set. Even this deviation may be still acceptable for practice. In Figure 3.6, the 10%, 50% and 100%-approximation sets for test instance 3 are plotted. It can

[11] It can be observed that for test instance 2, the hypervolume of the solutions delivered by the APS algorithm is slightly larger in the average than that of the reference set $B$. If $B$ would be the exact Pareto front, this would not be possible, but it should be noted that during the determination of $B$ by CE, objective function values for $h(x)$ have been estimated by sampling as well (although with a large sample size), such that neither all points in $B$ need to be Pareto-optimal indeed, nor is it guaranteed that the objective function estimates are exact.

| t.c. | type | B | APS | | | |
|---|---|---|---|---|---|---|
| | | $I_H$ | $\mu(I_H)$ | $\mu(I_H^-)$ | $\mu(I_{\epsilon+}^1)$ | $\mu(I_\epsilon^1)$ |
| 1 | etc | 0.2850 | 0.2849 | 0.0002 | 0.0023 | 1.0020 |
| 2 | elc | 0.3845 | 0.3847 | -0.0002 | 0.0024 | 1.0019 |
| 3 | elu | 0.2842 | 0.2838 | 0.0004 | 0.0039 | 1.0036 |
| 4 | etc | 0.3056 | 0.3054 | 0.0002 | 0.0040 | 1.0036 |
| 5 | etu | 0.2415 | 0.2409 | 0.0006 | 0.0030 | 1.0027 |
| 6 | vtc | 0.6248 | 0.6247 | 0.0002 | 0.0019 | 1.0015 |
| 7 | vlc | 0.1446 | 0.1445 | 0.0001 | 0.0024 | 1.0016 |
| 8 | vlu | 0.3925 | 0.3921 | 0.0003 | 0.0031 | 1.0025 |
| 9 | vtc | 0.3212 | 0.3162 | 0.0050 | 0.4684 | 1.3517 |
| 10 | vtu | 0.4676 | 0.4669 | 0.0007 | 0.0026 | 1.0019 |

Table 3.7.: Mean values of selected performance measures over 10 runs for 10 random test instances.

be seen that the worst-case performance is close to the best-case performance. Thus, the algorithm is stable with respect to the random decisions made during optimization at this test instance.

The decision maker can use the Pareto-optimal set of a problem to identify the solution that fits his needs. As an example, in Table 3.9, we present the solution set of test instance 6. Now, the objective function values are given in their original (not re-scaled) form; $h(x)$ is multiplied by $-1$ to make the values positive. The elements of the solution set are indexed by 0 to 8. Solution 8 maximizes the weighted average $fc(y,x)$ of strategic and economic gains, but this comes at the price of a relative high expected overtime cost $h(x)$, which indicates that this portfolio is not robust. Solutions 1 to 7 may be interesting portfolios for a more risk-averse decision maker. S/he can consider the detailed properties of each of these portfolios and make then a final choice. Solution 0 is the empty portfolio which is not of practical interest.

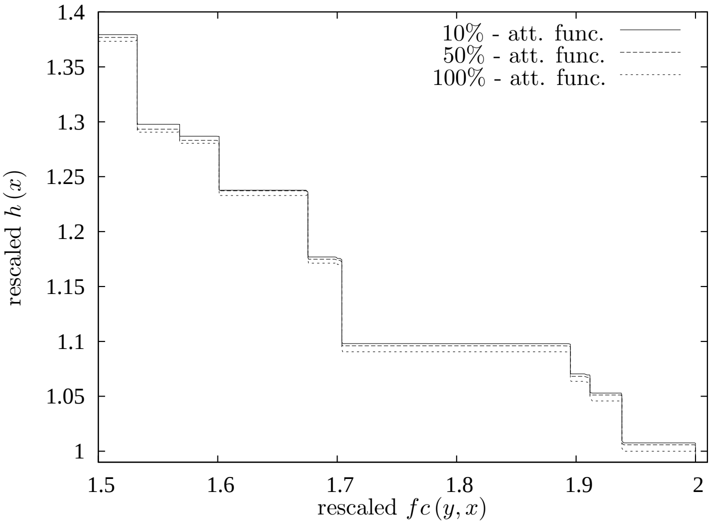



Figure 3.6.: $k$%-approximation sets for test instance 3.

**Synergies and Cannibalization.** In general synergies and cannibalization effect between different projects may occur. Synergy means that the benefit yielded by the implementation of two or more projects together is larger than the sum of the single benefits, whereas in the case of cannibalization the benefit of combinations of projects is smaller than the than the sum of the single benefits. As stated above, in our special real-world application, synergy and cannibalization effects did not play a role, such that $\mathbb{E}(w_{is}^{(1)})$ was set to zero. In order to test our procedure also for the situation of non-vanishing quadratic terms in the objective function $fc(y,x)$, we modified the ten test cases described above by introducing (positive or negative) terms $\mathbb{E}(w_{is}^{(1)})$ in the following way: For each $i < s$, a random factor $\zeta$ was drawn from a uniform distribution on $[-1,2]$, and $\mathbb{E}(w_{is}^{(1)})$ was set to the value $\zeta \cdot (\mathbb{E}(w_i^{(1)}) + \mathbb{E}(w_s^{(1)})$. Computationally, the extension to terms that are quadratic in the $y_i$ variables does not cause any difficulties as the master problem is solved by the NSGA-II algorithm which does not require linearity. The result is shown in Table 3.8. It can be observed that the achieved solution quality is comparable to that in the linear case (Table 3.7).

| t.c. | B | APS | | | |
|---|---|---|---|---|---|
| | $I_H$ | $\mu(I_H)$ | $\mu(I_H^-)$ | $\mu(I_{\epsilon+}^1)$ | $\mu(I_\epsilon^1)$ |
| 1 | 0.6639 | 0.6637 | 0.0001 | 0.0012 | 1.0007 |
| 2 | 0.6588 | 0.6588 | 0.0000 | 0.0008 | 1.0005 |
| 3 | 0.7202 | 0.7196 | 0.0006 | 0.0014 | 1.0008 |
| 4 | 0.6527 | 0.6523 | 0.0004 | 0.0021 | 1.0013 |
| 5 | 0.6878 | 0.6874 | 0.0004 | 0.0010 | 1.0006 |
| 6 | 0.8459 | 0.8459 | 0.0000 | 0.0011 | 1.0006 |
| 7 | 0.8481 | 0.8477 | 0.0004 | 0.0009 | 1.0005 |
| 8 | 0.8397 | 0.8378 | 0.0019 | 0.1583 | 1.0859 |
| 9 | 0.7237 | 0.7239 | -0.0003 | 0.0021 | 1.0013 |
| 10 | 0.8655 | 0.8656 | -0.0001 | 0.0015 | 1.0008 |

Table 3.8.: Mean values of selected performance measures over 10 runs for 10 random test instances, with quadratic $fc(y,x)$.

**Sensitivity Analysis.** Our model requires the estimation of a considerable number of parameters, which raises the question of robustness with respect to the accuracy of parameter estimates. Let us mention that a good part of these parameters (e.g.: expected gains, required efforts, ready dates, due dates, etc.) have to be determined or estimated also in a traditional form of project management. Whatever method of project planning the decision maker applies, the quality of the derived project portfolio and work plan will largely depend on the appropriateness of the estimates of these crucial parameters, and concerning them, the proposed model does not introduce any additional difficulty. Nevertheless, there are some new parameters in our model that do not occur in current planning methods, and it is an interesting question how sensitive the provided solutions are with respect to their estimates. We generated an additional test instance (with synergy and cannibalization terms) and performed a sensitivity analysis to determine the influences of the parameters $Var(U_i)$ and $\gamma_{rj}$. First, we varied the variances $Var(U_i)$ of the random variables $U_i$, multiplying the variance of each $U_i$ by a factor uniformly drawn from $[0.8, 1.2]$. This was repeated ten times, such that we obtained ten "mutants" of the "true" test instance. In this way, a situation is modelled where the decision maker is able to provide an unbiased estimate of the effort, but is inaccurate with her/her his estimation of the degree of randomness. (We did not consider changes of the expected values of the efforts, since each planning technique must necessarily be sensitive with respect to a bias in effort estimates.) It turned out that a change of the variance of this magnitude left the solution rather robust: in the average over the ten mutants, 68% of the efficient portfolios remained invariant, while, of course, the corresponding $h(x)$ values slightly changed.

| label | $y$ | $fc\,(y,x)$ | $h\,(x)$ | 95% confidence interval for $h\,(x)$ |
|---|---|---|---|---|
| 0 | 000000000000 | 0.00 | 0.00 | $[0.00, 0.00]$ |
| 1 | 100110000001 | 239.92 | 70.95 | $[70.79, 71.10]$ |
| 2 | 100110100010 | 294.68 | 73.89 | $[73.76, 74.02]$ |
| 3 | 100110100011 | 303.59 | 73.95 | $[73.83, 74.07]$ |
| 4 | 100010110011 | 321.98 | 76.95 | $[76.80, 77.11]$ |
| 5 | 010100110011 | 353.55 | 77.01 | $[76.85, 77.17]$ |
| 6 | 100100110011 | 373.20 | 78.97 | $[78.79, 79.15]$ |
| 7 | 100101110011 | 376.29 | 80.08 | $[79.94, 80.22]$ |
| 8 | 000101110011 | 383.32 | 188.73 | $[188.28, 189.17]$ |

Table 3.9.: Set of Pareto-optimal solutions of test instance 6.

Similarly, if (in each of ten mutants) each of the values $\gamma_{rj}$ was multiplied by a factor randomly selected from $[0.8, 1.2]$, in the average, 72% of the efficient portfolios remained invariant.

Table 3.10 shows the effect of the estimation errors measured by hypervolumes and epsilon indicators. Column $I_H$ of $B$ contains the hypervolume of the true efficient frontier, determined by complete enumeration for the original test instance. In columns $\mu(I_H)$ etc. for $CE$, the complete enumeration solutions for the ten mutated instances are treated as if they were proposed-efficient solutions delivered by a heuristic for the original instance, and evaluated by the metrics used in Tables 3.7 and 3.8. We see that the fit is still rather good despite of the assumed estimation errors. The solution quality appears to be more sensitive with respect to estimation errors concerning effort variances compared to errors concerning employee efficiency values.

| disturbed parameter | B | CE | | | |
|---|---|---|---|---|---|
| | $I_H$ | $\mu\,(I_H)$ | $\mu\,(I_H^-)$ | $\mu\left(I_{\epsilon+}^1\right)$ | $\mu\left(I_\epsilon^1\right)$ |
| $Var\,(U_i)$ | 0.66160 | 0.66124 | 0.00033 | 0.00565 | 1.00353 |
| $\gamma_{rj}$ | 0.66160 | 0.66149 | 0.00008 | 0.00091 | 1.00056 |

Table 3.10.: Estimation errors measured by selected performance measures over 10 runs for 10 random test instances.

## 3.7. Concluding Remarks

In this Chapter we presented a multi-objective model for project portfolio selection that considers both economic and competence-oriented goals, and a bi-objective version of the model under uncertainty. Competency gains along certain desirable profiles are used to formulate competence-oriented goals. In addition to the different skill sets of employees', learning and knowledge depreciation effects are included. In the stochastic version of the model a third type of objectives is considered that measures the robustness of a certain portfolio in terms of expected surplus costs due to overtime or external work. We developed a *linear* (stochastic) (mixed-integer) multi-objective program, that approximates the original problem. The approximation allows the exact solution of the problem related to the the assignment of available personnel to work packages of the selected projects over time, i.e., over the single periods of a planning interval, by using an LP solver. For the solution of the "rich" discrete portfolio optimization master problem we propose to apply a multi-objective metaheuristic technique. In the deterministic setting we have investigated two metaheuristics for the described purpose: the NSGA-II algorithm that builds on the Genetic Algorithms paradigm, and the P-ACO algorithm that makes use of the Ant Colony Optimization approach. We tested our proposed methods on two sets of test instances: randomly generated synthetic test cases of different size and type, as well as a real-world application delivered by the E-Commerce Competence Center Austria. In our synthetic test instances, the NSGA-II approach outperformed P-ACO. For the real-world instances, a slight superiority of P-ACO and of NSGA-II in the case of lower and of higher invested computation times, respectively, could be stated. Both techniques provided reasonable solutions from a practical point of view.

To solve the stochastic problem we designed a procedure based on the APS (Adaptive Pareto Sampling) technique in combination with the aforementioned NSGA-II algorithm, and obtained experimental results on a series of test instances derived from a real-world application case. Well-known performance indicators for the evaluation of multi-objective heuristics have been used to assess the quality of the results. For all test instances except one, the proposed technique turned out to perform practically equally well as an approach combining complete enumeration with extensive simulation, although consuming only 1% of the runtime of the last-mentioned approach; even in the case of the exceptional test instance, the deviation of the solution quality is less than 1.6%. Concluding from these results, we anticipate that our technique is well-suited also for solving test instances for which complete enumeration is not a feasible option anymore.

Our work has shown the desirability of future research in several directions. Let us outline six topics where future research will be particularly helpful. First, collecting data for determining the model parameters has proved as a rather time-consuming task. Suitable tools and integrated semi-automatized systems should be developed to support this process. Moreover, also methodological questions concerning the estimation of competence scores and the relation of such scores to efficiencies deserve further attention.

Secondly, the model in Section 3.2 does not involve precedence relations between tasks or projects. For the single-objective version of the model, it has been shown in Gutjahr et al. [57] that the addition of precedence relations between tasks of a project does not violate the LP structure of the linearized problem. The same holds for our multi-objective problem formulation, and probably this observation can also be generalized to precedence relations between tasks of different projects or between projects. Therefore, the exact solutions for small instances can be determined as in Section 100 also in the case of precedence relations.

Third, the envisaged planning time horizon in our model is rather the medium term than the long term. For long term strategic planning (sometimes already for shorter planning periods), aspects as changes of personnel or possible subcontracting or outsourcing play a role. Our model could be extended by including these aspects. E.g., external costs by subcontracting or outsourcing could be treated along the lines of the approach in Heimerl and Kolisch [60].

Fourth, our model is static, i.e., it presupposes a fixed time horizon and a decision to be made only at the beginning of the considered period.[12] This planning paradigm excludes adaptive policies, and it is susceptible to end-of-horizon effects. An extension of the presented approach to a situation where the assignment of personnel to tasks over time is done in a *dynamic* way, depending on actual work times as they become gradually known during the execution of the selected projects would be as interesting as challenging. Some preliminary results (using dynamic-stochastic project scheduling approaches from the literature, cf., e.g., Möhring and Stork [91]) have been outlined in Gutjahr et al. [56], Reiter et al. [106], but much remains still do be done.

Fifth, attempts to solve instances of the considered deterministic problem by suitable *exact* methods should be made. In this context, Lagrangian duality approaches for problem decomposition (see, e.g., Lassiter et al. [80]) and hierarchical decomposition techniques for large-scale multi-objective systems, as developed in Caballero et al. [23], Tarvainen and Haimes [117], might be explored in future research.

Finally, we have presented the APS technique here within the context of a particular SMOCO (stochastic multi-objective combinatorial optimization) problem. However, it can be used with only slight adaptations also within a large class of other problems of SMOCO type. Future research should explore this potential. In principle, the consideration of

[12] In practice, it is often the case that portfolio planning is done in a rolling-horizon manner. In such a situation, our analytic approach can be applied anew each time a new decision is to be made, based on the currently available information. This "iterated static" decision process, however, is not yet a *dynamic* decision process that anticipates re-planning by changes.

more than two objective functions is possible as well, based on the same algorithmic framework. The application of the NSGA-II algorithm as an auxiliary procedure for providing APS with solution set candidates has turned out as successful in our experiments, but of course also other (metaheuristic or mathematically oriented) methods can be used for this purpose.

# 4. Application to Vehicle Routing

## 4.1. Problem Description

In this Chapter, we present a bi-objective extension of the classical capacitated vehicle routing problem (CVRP) and exact algorithms for solving the considered problem. In addition to the traditional objective, (i) minimization of total travel cost, we also consider a second objective, (ii) minimization of the length of the longest route. The CVRP can be defined as follows. The objective is to find optimal routes for a fleet of $K$ identical vehicles serving a set of $n$ customers and based at a single depot. Each customer $i = 1, \ldots, n$ has a deterministic demand $q_i$, that is known in advance. The fleet of vehicles is homogeneous, each vehicle having a maximum capacity $Q$ it can deliver. A feasible solution for the CVRP is represented by a set of routes, each starting and ending at the depot and satisfying the conditions that (i) each customer is visited exactly once and (ii) the total demand of the customers on each route is at most $Q$. Nonnegative costs $c_{ij}$, representing the travel cost needed to drive from customer $i$ to customer $j$, are associated with each pair of customers $(i, j)$. The objective is minimize the total cost, while serving all customers. As a generalization of the traveling salesman problem (TSP), the CVRP is $\mathcal{NP}$-hard. A vast literature dealing with the CVRP exists, including articles presenting a large number of solution methods in the fields of exact methods, problem-specific heuristics and meta-heuristic algorithms. An overview has been given in Toth and Vigo[118]. Considering exact methods, branch-and-cut algorithms are among the best currently available solution techniques for the CVRP (see, e.g., Baldacci et al.[14], Fukasawa et al.[46], Lysgaard et al.[86]). Taking into account real-life application, total travel cost is often not the only measure to assess the quality of a solution. Different other aspects are present. Especially the distribution of the driver workload, i.e., the balance of route lengths, is another important measure. To deal with the requirement of "sufficiently balanced" routes, two different approaches are possible: (i) introducing hard constraints, or (ii) an additional objective function taking account of balance. From the second approach, we obtain bi-objective problem, denoted as CVRP with route balancing (CVRPB). (An overview on multi-objective vehicle routing problems is given in Jozefowiez et al.[72].) To express the route balancing objective, different formulations can be used. Natural choices are e.g. (i) minimization of the length of the longest route or (ii) minimization of the difference between the lengths of the longest and the shortest route (cf. Jozefowiez et al.[70, 71], Pasia et al.[95,96]). Several heuristic solution methods exist to solve CVRPBs (see Jozefowiez et al.[72]), but, to our knowledge, no algorithm that determines the exact Pareto front has been described up to now.

Among the aforementioned variants we address the first formulation, where the length of the longest route represents the second objective function. In general, there will be no single solution that attains the optimum of both objectives at the same time. Therefore, it is desirable to compute the set of Pareto-optimal (or: efficient) solutions (short *Pareto set*).[1] Classical methods for determining the Pareto set are, e.g., extensions of the weighted sum method, $\varepsilon$-constraint methods or weighted metric methods.[2]

We use the adaptive $\varepsilon$-constraint method by Laumanns et al.[81] in combination with a branch-and-cut algorithm and two genetic algorithms (GAs), namely a single-objective GA and the multi-objective NSGA-II (Deb et al.[33]), to solve the considered problem. The adaptive $\varepsilon$-constraint method determines the Pareto set by solving a sequence of constrained single-objective problems. In our implementation, the second objective function is treated as a constraint. This leads to a distance-constrained CVRP (short: DCVRP). An efficient branch-and-cut algorithm is used to solve the DCVRP. The GAs are applied to generate good incumbent candidates for the branch-and-cut algorithm in order to speed up the search process. They are called either in a sequential way (NSGA-II) or in an interactive way (single-objective GA).

Instead of a straightforward three-index problem formulation providing a special index for the vehicle under consideration, we apply a more efficient two-index formulation proposed by Laporte et al.[78,79] for the DCVRP, which, however, is not linear anymore in the case of the CVRPB. Nevertheless, by the specific way we organize the $\varepsilon$-constraint algorithm, the resulting subproblems of DCVRP type become linear. The problem formulation requires the computation of valid and efficient lower bounds for a multiple traveling salesman problem. We apply generalized Held-Karp bounds for this purpose (a technique that could also be used in the single-objective DCVRP case). Moreover, our algorithm ensures that cuts that are applied in the branch-and-cut solution process in one of the iterations remain valid in the subsequent iterations, which can be exploited to improve performance.

[1] A solution $x^{(1)}$ is called Pareto-optimal if there is no other solution $x^{(2)}$ that has is at least as good as $x$ in all objectives, and strictly better than $x$ in at least one objective.

[2] An introduction to multi-objective optimization is given in Section 2.1.

This chapter is organized as follows. Section 4.2 presents the mathematical model of the problem. In Section 4.3.1, we specify the different components of our algorithms. In Section 4.5, the efficiency of the new algorithm on a set of standard CVRP benchmark instances from the TSPLIB is assessed. Section 4.6, finally, gives concluding remarks.

## 4.2. Model Formulation

This section describes our model for the CVRPB. It can be seen as an extension of the model used by Laporte et al. [78,79] and by Achuthan et al. [1] for the DCVRP. We assume that travel time matrix and cost matrix coincide and denote this matrix by $C$. It is assumed that $C$ is symmetric and that no service times are present. The elements of $C$ are supposed to fulfill the triangle inequality (i.e., the distance function is a metric). The CVRPB with (i) minimization of the total cost and (ii) minimization of the distance of the longest route can then be formulated as follows. The problem is defined on a undirected graph $G = (V, E)$, where $V = \{0, 1, \ldots, n\}$ is the set of vertices, and $E = \{\{i, j\} : i, j \in V,\ i < j\}$ is the set of edges. Index 0 denotes the depot, where $K$ vehicles of capacity $Q$ and maximum allowable route length $D$ are located. The set of customers is given as $V_0 = V \setminus \{0\}$. Each customer $i$ has a nonnegative demand $q_i$. Furthermore, to each edge $e \in E$, a cost value $c_e$ is associated, which can also be interpreted as the travel time or as the length of edge $e$.

For abbreviation, $\delta(S)$ denotes the set of edges in $G$ with exactly one end-vertex in $S$, i.e., $\delta(S) = \{\{i, j\} \in E : i \in S, j \in V \setminus S\}$, and $\delta(\{i\})$ is shortly written as $\delta(i)$. Moreover, $\gamma(S)$ is the set of edges with both ends in $S$, i.e., $\gamma(S) = \{\{i, j\} \in E : i, j \in S\}$. Finally, $\left(S : \bar{S}\right)$ denotes the set of edges with one end vertex in $S$ and the other in $\bar{S}$. For each edge $e$, the decision variable $x_e$ is defined as the multiplicity of edge $e$ being used as part of a route, where $x_e \in \{0, 1\}$ if $e$ is not incident with the depot, and $x_e \in \{0, 1, 2\}$ otherwise.

The considered CVRPB is given by the following nonlinear bi-objective optimization problem, which extends the DCVRP formulation in Laporte et al. [78,79] to the bi-objective case. The problem formulation contains the expression $\bar{r}(S, D)$ which will be defined below.

$$
\begin{aligned}
\text{(CVRPB)} \min \quad & \left( \sum_{e \in E} c_e x_e,\ D \right) && && (4.1)\\
\text{s.t.} \quad & \sum_{e \in \delta(i)} x_e = 2 && \forall i \in V_0, && (4.2)\\
& \sum_{e \in \delta(0)} x_e = 2K, && && (4.3)\\
& \sum_{e \in \delta(S)} x_e \geq 2\bar{r}\,(S, D) && \forall S \subseteq V_0, |S| \geq 2, && (4.4)\\
& x_e \in \{0, 1\} && \forall e \notin \delta\,(0)\,, &&\\
& x_e \in \{0, 1, 2\} && \forall e \in \delta\,(0)\,, &&\\
& D \geq 0. && && (4.5)
\end{aligned}
$$

Equation (4.1) defines the two objective functions to be minimized. Equations (4.2) and (4.3) assure that exactly two edges are incident to each customer vertex and that exactly 2K edges are incident to the depot vertex.

The capacity cut constraints (4.4) impose that each route is connected to the depot, and that for each route, capacity restrictions as well as distance restrictions are respected. Therein, the expression $\bar{r}\,(S, D)$ denotes a lower bound on the number of vehicles needed to serve all the customers in $S$ in the optimal solution of the entire problem. This number is calculated as the maximum of two expressions $r_1\,(S)$ and $r_2\,(S, D)$ which are defined as follows:

$r_1\,(S)$ is the optimal solution to the Bin Packing Problem (BPP) with capacity $Q$ and item sizes given by the demands $q_i$ of the customers $i \in S$. It is well-known that in capacity cut constraints, $r_1\,(S)$ can be replaced by the trivial BPP lower bound $k_1\,(S) = \lceil q\,(S)/Q \rceil$, where $q\,(S)$ is the total demand of all customers in $S$.

$r_2\,(S, D)$ is the minimal integer $v$ that satisfies the equation

$$
v = \left\lceil \frac{H_v\,(S)}{D} \right\rceil,\ v = 1, \ldots, |S|, \tag{4.6}
$$

where $H_v\,(S)$ is the optimal value of the multiple traveling salesman problem (m-TSP) solution with a fixed number $v$ of salesmen visiting all customers in the set $S$ and starting and ending at the depot.

Eq. (4.6) is a fixed-point equation, and defining $r_2\,(S, D)$ as the smallest integer satisfying (4.6) it is only correct if it is guaranteed that (4.6) has at least one solution. To show this, let us set $\varphi(v) = \lceil H_v(S)/D \rceil$ and $\psi(v) = \varphi(v) - v$, and assume $H_K(\{1, \ldots, n\}) \leq KD$ (which is a necessary condition for the feasible set being nonempty). Then we obtain $\varphi(1) \geq 1$ and $\varphi(K) \leq K$ or $\psi(1) \geq 0$ and $\psi(K) \leq 0$. Using the triangle inequality, one verifies that $\varphi(v)$ is nondecreasing in $v$, such that $\psi(v+1) \geq \psi(v) - 1$ for all $v$. Therefore, $\psi$ must have a root $v$, i.e., an argument value $v$ where $\psi(v) = 0$. In total, this ensures that the smallest root of $\psi$ exists, i.e., that $r_2(S, D)$ is well-defined.

The m-TSP is known to be $\mathcal{NP}$-hard, but it can be shown that it is allowed to use a lower bound for the m-TSP solution value instead of $r_2(S, D)$ (see section 4.3.2).

By fixing $D$, it is easy to see that (4.4) is a valid inequality for each Pareto-optimal solution of (CVRPB). Furthermore, note that eq. (4.4) excludes routes of a length larger than $D$. We show this by contradiction: Assume that in a Pareto-optimal solution of (4.2) – (4.5), a route with a length larger than $D$ occurs. Let $S$ be the set of customers on this route. The route must already be of minimal length on $S$, because otherwise, by a re-arrangement of the customers on the route, the first objective function value could be reduced without reducing the second objective function value, such that the solution would not be Pareto-optimal. Thus, since $H_1(S)$ is the optimal solution value of the ordinary TSP with customer set $S$, the considered route has length $H_1(S)$. Therefore, $H_1(S) > D$ or $\lceil H_1(S)/D \rceil > 1$, and hence $r_2(S, D) \geq 2$ by eq. (4.6). However, for the considered set $S$, the l.h.s. of (4.4) is equal to 2, which yields the contradiction $2 \geq 4$.

Equations (4.5), finally, restrict the values of the decision variables to their feasible ranges.

In view of the definition of $r_2(S, D)$ by eq. (4.6), the problem (CVRP) is nonlinear, considering that $D$ is a decision variable. We shall remove this nonlinearity by the algorithmic approach described in the next section, where bounds on $D$ will be introduced.

Let us emphasize that by constraint (4.3), the number of routes is always fixed to the pre-defined value $K$, i.e., we assume that a fixed fleet size is given and each vehicle has to be used.

## 4.3. Solution Techniques

### 4.3.1. General Approach

For identifying all Pareto-optimal solutions of the problem (CVRPB) defined above, an exact algorithm capable of solving multi-objective combinatorial optimization (MOCO) problems within reasonable time is required. Most algorithms for multi-objective optimization problems use a sequence of parametrized single-objective problems to find

Pareto-optimal solutions. Such algorithms select a scalarization method producing single-objective subproblems, provide an appropriate scheme to vary the parameters of the scalarization, and apply a solver that guarantees to find the optimal solutions of the subproblems. Although MOCO problems have a finite number of Pareto-optimal solutions, some of the traditional algorithms are not capable of generating all solutions. For example, the weighted sum approach only guarantees to find supported solutions (solutions that lie, in the objective space, on the convex hull of the Pareto front). Algorithms based on weighted metrics overcome this problem e.g. the WCN algorithm by Ralphs et al.[104]. The traditional $\varepsilon$-constraint method (Haimes et al.[59]) uses a predefined grid over the objective space. The complete Pareto set can only be identified if each cell contains at most one Pareto-optimal solution. The adaptive $\varepsilon$-constraint method Laumanns et al.[81] overcomes this disadvantage by varying the parameters of a single-objective problem in such a way that all Pareto-optimal solutions can be found by solving $O(\kappa^{d-1})$ single-objective problems, where $d$ is the number of objectives and $\kappa$ is the cardinality of the Pareto set.

To ensure that the solution of the single-objective subproblem is optimal, we use a branch-and-cut algorithm. Different branch-and-cut algorithms were successfully applied to CVRP problems, e.g., the algorithm implemented by Lysgaard et al.[86]. The separation routines of their implementation are available at (Lysgaard[85]). Despite the large interest in exact algorithms for the CVRP, exact methods for the DCVRP received comparably little attention in the literature. To our knowledge, no new exact algorithm for the DCVRP has been presented since the articles by Laporte et al.[78,79]. In our implementation of a branch-and-cut algorithm for the DCVRP, we use the separation routines proposed in (Lysgaard[85], Lysgaard et al.[86]) to treat the capacity constraints, and, in addition, we have implemented separation routines to identify violated distance constraints.

For branch-and-cut algorithms, finding feasible solutions at early stages of the solution process plays a major role. The overall computational effort can be reduced in this way, and the search is guided to promising regions of the solution space. We apply meta-heuristic algorithms for identifying good feasible solutions. Two different hybridizations with heuristic algorithms are possible: (i) a sequential combination, or (ii) an interactive combination. In the sequential combination, a multi-objective optimizer (MOO) is run before the adaptive $\varepsilon$-constraint method starts. In each iteration of the adaptive $\varepsilon$-constraint method, a solution of the non-dominated set proposed by the MOO is used as an incumbent candidate. In the interactive combination, in each iteration of the adaptive $\varepsilon$-constraint method, a single objective optimizer is called to generate an incumbent candidate.

In the sequential combination, we use the multi-objective genetic algorithm NSGA-II for two reasons: First, NSGA-II belongs to the currently best-performing multi-objective metaheuristics. Secondly, an NSGA-II based algorithm has already been successfully applied to the CVRPB by Jozefowiez et al.[70,71]. Let us mention that good results in the heuristic solution of the CVRPB have also been obtained by Population-Based Local Search (Pasia et al.[95]) and by Pareto Ant Colony Optimization (Pasia et al.[96]); an extension of our approach to these techniques is easily possible. In the sequel, the combination of the $\varepsilon$-constraint method with NSGA-II will be denoted by EPSN. In the interactive combination, a single-objective GA capable of solving DCVRPs is used to generate incumbent candidates. This combination is denoted by EPSS. A general description of the $\varepsilon$-constraint method, as well as of the NSGA-II algorithm is given in Section 2.4.

For our test instances, we set $\Delta = 1$, which is allowed since all objective function coefficients in these instances are integer [3]. In our case, the vector $x$ of decision variables is given by $(x, D)$, where $x$ contains the variables $x_e$. Moreover, $f_2(x, D) = D$, such that the first constraint in (2.9) becomes $D \leq \varepsilon_2 - \Delta$. The set $X$ consists of all elements $(x, D)$ satisfying the constraints (4.2) – (4.5).

We show that the solution of $\min f_1(x)$ under the constraints $D \leq \varepsilon_2 - \Delta$ and (4.2) – (4.5) is equivalent to the solution of

$$\begin{aligned} &\min f_1(x) \\ &\text{s.t.} \sum_{e \in \delta(S)} x_e \geq 2\bar{r}(S, \varepsilon_2 - \Delta) \ \forall S \subseteq V_0, \ |S| \geq 2 \text{ and} \\ &\qquad (4.2), (4.3) \text{ and } (4.5). \end{aligned} \tag{4.7}$$

In this formulation, we have replaced the $D$ in eq. (4.4) by the upper bound $\epsilon_2 - \Delta$ that the $\epsilon$-constraint algorithm fixes for $D$ in a current iteration. This is only allowed if it can be ensured that the function $r_2(S, D)$, and therefore also the function $\bar{r}(S, D)$, is nonincreasing in $D$; otherwise, it could happen that for some $D$ smaller than $\epsilon_2 - \Delta$, constraint (4.4) would be weaker for this $D$ than for $\epsilon_2 - \Delta$, with the effect that problem (4.7) would over-estimate the true minimum. In the sequel, we show that this cannot occur: Assume $S$ as fixed, and let $\psi_D(v) = \lceil H_v(S)/D \rceil - v$. If $D' \geq D$, we obtain $\psi_{D'}(v) \leq \psi_D(v)$. Therefore, the smallest value $v$ for which the function $\psi_{D'}$ vanishes is smaller or equal to the smallest value of $v$ for which $\psi_D$ vanishes. This shows $r_2(S, D') \leq r_2(S, D)$. As a consequence, the validity of (4.4) for $D$ implies the validity of (4.4) for $D'$, i.e., with $X_D$ denoting the set of all $x$ such that (4.2) – (4.5) hold for fixed $D$, we have $X_D \subseteq X_{D'}$

[3] In cases where the greatest common divisor of all coefficients is larger than one, $\Delta$ can be set to this larger value, cf. Bérubé et al.[18].

and therefore $\min_{x \in X_{D'}} f_1(x) \leq \min_{x \in X_D} f_1(x)$. Thus, the minimum value for $f_1(x)$ is obtained by making $D$ as large as possible, i.e., by setting $D = \varepsilon_2 - \Delta$.

The optimization problem (4.7) is a DCVRP. The expression $\bar{r}(S, \varepsilon_2 - \Delta)$ is a constant now, which means that an integer linear program has been obtained.

### 4.3.2. Branch-and-Cut

Branch-and-cut solves a sequence of linear programming (LP) relaxations of an integer program (IP). After solving the LP relaxation, if the current node cannot be pruned, *cutting planes* are added to the problem. If no violated cuts are found and the current solution is not integer feasible, new nodes are created by branching. The performance of a branch-and-cut algorithm depends on the quality of the bounds needed to prune nodes, the procedures to find violated cuts (separation procedures), and the branching strategy.

In our experiments, it turned out as advantageous to start the branch-and-cut algorithm with an LP relaxation including only the degree constraints (4.2) and (4.3). Violated cuts induced by the capacity and distance restrictions are added as needed.

#### Separation Procedures related to Capacity Constraints

To introduce cuts concerning capacity constraints, we use the CVRPSEP package (Lysgaard [85]). This package includes separation routines for capacity, framed capacity, strengthened comb, multistar, partial multistar, generalized multistar and hypotour cuts described in Lysgaard et al. [86].

#### Separation Procedures related to Distance Constraints

To find violated cuts for the distance constraints, we implemented a method to get a lower bound for the m-TSP and four specific separation procedures building on it.

For further use, let us start with the following definitions. Let $x^*$ be the current LP solution, then $G^* = (V, E^*)$ with $E^* = \{e \in E : x_e^* > 0\}$ is the so-called support graph. By $G_0^* = (V_0, E_0^*)$, we denote the graph obtained from $G^*$ by removing the depot and the edges incident with the depot.

A violated distance cut is found if for a subset $S \subseteq V_0$, the function

$$f(S) = \sum_{e \in \delta(S)} x_e^* - 2v_0 \tag{4.8}$$

takes a negative value, where $v_0$ is any lower bound on $r_2(S, D)$ (note that in this case, eq. (4.4) cannot be satisfied anymore). As shown by Achuthan et al. [1], the bound $v_0 =$

$\left\lceil \sum_{e\in\delta(S\cup\{0\})} c_e x_e / D \right\rceil$ used in Laporte et al.[78,79] may fail when some capacity constraints are violated and/or the current solution is not integer feasible. Achuthan et al. suggest to use a valid lower bound for the objective function value of the corresponding m-TSP, but they do not carry out this approach in Achuthan et al.[1]. We adopt their suggestion by generalizing the well-known Held-Karp lower bound (Held and Karp[62]) for the TSP to case of the m-TSP, applying the Held-Karp bound to an extended graph. This will yield a lower bound $\chi_v(S)$ for $H_v(S)$. Although the estimate constitutes a valid lower bound for the r.h.s. of (4.4) and can therefore be added as a cut, it may eventually be too weak to exclude routes that violate the distance constraint (as shown in section 4.2, the *exact* m-TSP solution *does* exclude such routes). Therefore, we have to rely on an additional separation procedure described below (separation of infeasible paths) which ensures the validity of the solutions by guaranteeing that the distance constraint is satisfied.

**(I) Lower Bound for the m-TSP** In order to extend the Held-Karp bound to the m-TSP, we apply a well-known reduction of the m-TSP to the TSP based on the introduction of pseudo-depots. The symmetric m-TSP is defined on a graph $G = (V, E)$, where $V = \{0, 1, \ldots, n\}$ is the set of vertices, and $E = \{\{i, j\} : i, j \in V,\, i < j\}$ is the set of edges. Index 0 denotes the depot, where $m$ vehicles are located. To edge $e \in E$, a cost (or travel time) value of $c_e$ is associated. The m-TSP consists in finding routes for all vehicles, each starting and ending at the depot and visiting each customer exactly once, such that the total cost of visiting all nodes is minimized. (An overview on techniques to solve the m-TSP is given in Bektas[16].) The m-TSP is reduced to a TSP on an extended graph $\bar{G} = \left(\bar{V}, \bar{E}\right)$ as follows: Add a set of $m-1$ vertices $\tilde{V} = \{n+1, n+2, \ldots, n+m-1\}$ to the vertex set of graph $G$ to obtain vertex set $\bar{V} = V \cup \tilde{V}$. The new nodes are considered as copies of the depot 0 or as "pseudo-depots". Extend the set of edges by setting $\bar{E} = E \cup \left\{\{i, j\} : i \in V, j \in \tilde{V}\right\}$. Assign to each new edge $\{i, j\}$ $(i \in V, j \in \tilde{V})$ a cost as follows: An edge between the depot 0 and one of the new nodes receives cost $\infty$. To an edge $\{i, j\}$ that connects a customer node $i$ to a new node $j$, the cost value of the edge $\{0, i\}$ is assigned. Then, the m-TSP on $G$ is equivalent to the TSP on $\bar{G}$.

For $\bar{G}$, we calculate a lower bound $\Gamma(\bar{G})$ for the Held-Karp bound, using the algorithm proposed by Valenzuela and Jones[121]. By the consideration above, $\Gamma(\bar{G})$ is then also a lower bound for the m-TSP solution value on $G$.

The algorithm by Valenzuela and Jones is based on the iterative estimation approach by Held and Karp[62], a Lagrangian relaxation method working with 1-trees and applying a perturbation determined by a set of weights $\pi_i$ that are assigned to the vertices $i = 0, \ldots, n$

of the complete graph. The weights are used to force the vertex degrees $d_i^T$ of an 1-tree[4] $T$ to a value of 2.The algorithm iteratively produces minimum 1-trees which increasingly resemble routes[5]. Given original edge lengths $c_e$, a modified length of an 1-tree is calculated by summation over the modified edge lengths $\hat{c}_e = c_e + \pi_i + \pi_j$, where $i, j$ are the indices of the vertices incident with edge $e$. Let $\hat{C} = (\hat{c}_e)$. Denoting the set of all 1-trees by $U$ and the set of all routes by $U_0$, we have $U_0 \subseteq U$, as every route is a 1-tree. With $L(C, T)$ and $L(\hat{C}, T)$ representing the length of the 1-tree $T$ under $C$ and $\hat{C}$, respectively, it is immediately seen that $L(\hat{C}, T) = L\,(C, T) + \sum_{i=0}^{n} d_i^T \pi_i$. In particular, if $T$ is a route, then $L(\hat{C}, T) = L\,(C, T) + \sum_{i=0}^{n} 2\pi_i$. In the case of a minimal-length route $T^*$, for every $\pi = (\pi_0, \ldots, \pi_n)$,

$$\min_{T \in U} L(\hat{C}, T) \leq \min_{T \in U_0} L(\hat{C}, T) = L(\hat{C}, T^*)$$

or

$$\min_{T \in U} \left\{ L(C, T) + \sum_{i=0}^{n} d_i^T \pi_i \right\} \;\leq\; L(C, T^*) + \sum_{i=0}^{n} 2\pi_i,$$

which yields

$$\Gamma_\pi = \min_{T \in U} \left\{ L\,(C, T) + \sum_{i=0}^{n} \left(d_i^T - 2\right) \pi_i \right\} \;\leq\; L\,(C, T^*)\,.$$

Thus, $\Gamma_\pi$ is a lower bound for $L\,(C, T^*)$. The Held-Karp bound results as $\max_\pi \Gamma_\pi$.

In the iterative framework, iterations $m = 0, \ldots, M$ are performed, where in each iteration the weights $\pi_i$ are changed. In our implementation, we used a schema proposed by Volgenant and Jonker [122]:

$$\pi_i^{(m+1)} = \begin{cases} \pi_i^{(m)}, & \text{if } d_i^{T^{(m)}} = 2, \\ \pi_i^{(m)} + b\, t^{(m)} (d_i^{T^{(m)}} - 2) + (1 - b)\, t^{(m)} (d_i^{T^{(m-1)}} - 2), & \text{otherwise.} \end{cases} \tag{4.9}$$

Therein, $\pi^{(0)} = (0, \ldots, 0)$, the symbol $T^{(m)} = T(\pi^{(m)})$ denotes the minimum 1-tree for the weight vector $\pi^{(m)}$, the constant $b \in [0, 1]$ is a parameter, and $t^{(m)}$ is the step length in the $m$-th iteration. As suggested by Valenzuela and Jones, the step length in iteration 0 is calculated as $t^{(0)} = L\,(C, T)\ /\ (2n)$, i.e., it is related to current average edge length. The value of $t^{(0)}$ is updated each time a better value for $\Gamma_\pi$ is found. In iteration $m$, the step length is computed as follows:

$$t^{(m)} = (m - 1) \left( \frac{2M - 5}{2\,(M - 1)} \right) t^{(0)} - (m - 2)\, t^{(0)} + \frac{(m - 1)\,(m - 2)}{2\,(M - 1)\,(M - 2)} t^{(0)}. \tag{4.10}$$

[4] A 1-tree is a minimum spanning tree on vertices $1, \ldots, n$ plus the two lowest cost edges connecting this tree with vertex 0.

[5] A route is a 1-tree with all vertex degrees equal to 2.

If the algorithm happens to generate a route, it is possible to use the corresponding $L(C,T)$ value as an upper bound and to stop the procedure as soon as the current lower bound exceeds this upper bound. If this does not occur, the procedure is stopped after the $M$-th iteration. This yields a lower bound $\Gamma_\pi = \Gamma(\bar{G})$ for the m-TSP solution value.[6]

In total, we obtain the procedure described in Algorithm 4.3.1 for computing a lower bound $w$ on the smallest integer $v = r_2(S,D)$ satisfying equation (4.6). To simplify notation, in Algorithm 4.3.1, we write $G$ instead of $\bar{G}$.

**Algorithm 4.3.1:** Procedure to calculate a lower bound on the number of vehicles needed to serve a set of costumers, considering distance constraints

**Input**: set of customers $S \subseteq V_0$, maximum distance $D$
initialize graph $G$ induced by set $S$;
set $w = 1$ and $stop = 0$;
**repeat**
  $u = \left\lceil \frac{\Gamma(G)}{D} \right\rceil$;
  **if** $u > w$ **then**
    add $u - w$ pseudo-depots to $G$, and set $w = u$;
  **else**
    $stop = 1$;
  **end**
**until** $stop = 1$;
**Output**: lower bound $w$ on smallest integer $v$ satisfying (4.6)

Let us verify that Algorithm 4.3.1 provides us with a lower bound on $r_2(S,D)$ indeed. First, note that by virtue of the reduction of the m-TSP to the TSP outlined above, $\Gamma(G)$ computes a lower bound $\chi_w(S)$ for the $w$-TSP solution value $H_w(S)$ on the graph induced by $S$. By adding only one depot (instead of $u - w$ depots) in each execution of the "then" branch in the algorithm, we would simply calculate the smallest (integer) root $w$ of $w = \lceil \chi_w(S)/D \rceil$. This would provide us with a lower bound on the solution of (4.6), since, with $\psi(w) = \lceil H_w(S)/D \rceil - w$ and $\bar{\psi}(w) = \lceil \chi_w(S)/D \rceil - w$, we have $\psi(w) \geq \bar{\psi}(w)$ for all $w$, such that the smallest root of $\bar{\psi}$ cannot be larger than the smallest root of $\psi$. All that remains to show is that whenever the obtained minimum number $u = \lceil \Gamma(G)/D \rceil = \lceil \chi_w(S)/D \rceil$ of required vehicles turns out as larger than the currently applied fleet size $w$, it is allowed to increase $w$ not only by one, but by $u - w$, skipping the values $w+1, w+2, \ldots, u-1$, as none of these values can be a root of $\psi$. We show this by contradiction: Assume that there exists some $u'$ with $w < u' < u$ such that (already)

[6] In our implementation, we chose $M = 300$ and $b = 0.6$.

$u'$ is the solution of (4.6), i.e., the smallest root of $\psi$. Then, since $H_w(s)$ is nondecreasing in $w$ in the considered case of a metric distance function,

$$u' = \left\lceil \frac{H_{u'}(S)}{D} \right\rceil \geq \left\lceil \frac{H_w(S)}{D} \right\rceil \geq \left\lceil \frac{\chi_w(S)}{D} \right\rceil = u,$$

which contradicts $u' < u$.

**(II) Specific Separation Procedures** To identify candidate sets of customers $S \subseteq V_0$, we use the following four procedures.

**Procedure 1: Check of Violated Capacity Cuts.** The first procedure checks all sets $S$ of customers for which the capacity cut separation routine of the CVRPSEP package indicated a violated capacity cut. For each of the these sets, the inequality

$$\sum_{e \in \delta(S)} x_e \geq 2 \max \left( \left\lceil \frac{q\,(S)}{Q} \right\rceil, w \right) \tag{4.11}$$

is added to the problem, where $w$ is the distance-related bound for $S$ calculated by Algorithm 3.2. Obviously, this cut strengthens a cut formulation only relying on the capacity constraints, i.e., on the first argument of the "max" function above.

**Procedure 2: Connected Components.** The implementations of this and the two following procedures are based on the separation routines used by Augerat et al. [12]. First, in order to improve the performance of the procedure, we apply a shrinking procedure to the graph $G^*$. Each edge with $x_e^* = 1$ is shrunk in the following way: The end nodes $i, j$ of $e$ are replaced by a super-vertex $k$ with demand $d_k = d_i + d_j$. Edges $\{i, \ell\}$ and $\{j, \ell\}$ are replaced by a single edge $\{k, \ell\}$ with edge value $x^*_{\{k,\ell\}} = x^*_{\{i,\ell\}} + x^*_{\{j,\ell\}}$. The shrinking procedure stops if no candidate edge for which $x_e^* = 1$ can be found. In each shrinking step, for the currently generated super-vertex, the Valenzuela-Jones bound for the value (4.6) is calculated by Algorithm 3.2, where $S$ is chosen as the set of original vertices contained in the super-vertex. With the obtained value $w$, a corresponding cut of the form (4.4) is added.

After this preparatory step, the connected-components procedure computes the connected components $S_1, \ldots, S_t$ of the resulting "shrunk" graph $G_0^*$. For each $i = 1, \ldots, t$, we check the distance constraint for $S_i$ as well as for $V_0 \backslash S_i$. In the case of violation, a cut of the form (4.4) is added.

**Procedure 3: Greedy Randomized Search.** The third procedure, which is only applied if no cut could be found by the former, is a simple greedy randomized search heuristic. It starts with a random initial set $S$ and adds, in each iteration, a customer $k$

to $S$ until $S$ contains all customers of the graph. Customer $k$ is chosen as the vertex $k$ for which $\sum_{e\in(S:\{k\})} x_e^*$ is maximum. For each set $S$ where $\sum_{e\in\gamma(S)} x_e^* - |S| + K > 0$, constraint (4.4) is checked.

As suggested by Augerat et al., if during the search a set $S$ is found for which $pD \geq \chi_v(S) \geq (p-\varepsilon) D$, where $p > 0$ and $\varepsilon$ with $0 < \varepsilon < 1$ are parameters, eq. (4.8) is checked for all sets $S \cup \{v\}$, where $v$ is adjacent to at least one node of $S$ in $G^*$.

**Procedure 4: Separation of Infeasible Paths.** In addition to the capacity cut constraints, cuts related to infeasible paths can be identified. This is a common technique used in branch-and-cut algorithms for routing problems, where the feasibility of the problem depends on the order of visits, e.g., routing problems with time windows Ascheuer et al.[8,9] and/or with packing constraints Tricoire et al.[119]. For convenience, a path $P$ consisting of edges $\{e = \{j_i, j_{i+1}\} \,|i = 1, \ldots, k-1\}$ is also written as $P = (j_1, j_2, \ldots, j_k)$. We assume that a path $P$ is always open and simple, i.e., $j_i \neq j_\ell$ for $i \neq \ell$. By $|P|$, we denote the number of edges on path $P$. We call a path $P$ infeasible if the length of the path plus the two distances of the start node and the end node, respectively, to the depot gives a value larger than the maximum allowed distance $D$. Formally, a path $P$ with start node $j_1$ and end node $j_k$ is infeasible if the expression

$$f(P) = \sum_{e\in P} c_e x_e + c_{\{0,j_1\}} + c_{\{0,j_k\}} - D \tag{4.12}$$

is larger than zero. For each infeasible path $P$, we can add the following constraint to the model in order to break $P$:

$$\sum_{e\in P} x_e \leq |P| - 1. \tag{4.13}$$

To identify infeasible paths, we use breadth-first search on the graph $G^*$, starting from each customer node. Algorithm 4.3.2 provides the pseudo-code of this search procedure. Therein, $P + \{j\}$ denotes the path obtained by appending, to path $P$, an edge from the last node of $P$ to node $j$. A similar algorithm is used by Tricoire et al.[119]. Evidently, if a solution contains a route violating the distance constraint, it must also contain an infeasible path, which will be recognized by the procedure above. Thus, the just-described procedure ensures the feasibility of the obtained solutions with respect to the distance constraint. Additionally, for each infeasible path we also check if the corresponding constraint (4.4) (using the Valenzuela-Jones lower bound) is violated. If a violation is found, we add (4.4) instead of the infeasible path constraint.

**Algorithm 4.3.2:** Procedure for finding infeasible paths

```
Input: a graph G* representing the current LP solution
set Q = ∅ and infPaths = ∅;
forall the i ∈ V_0 do
    Q = Q ∪ {i}, Paths[i] = {(i)};
end
while Q ≠ ∅ do
    i = select(Q), Q = Q \ {i};
    forall the P ∈ Paths[i] ∧ ¬processed(P) do
        if f(P) > 0 then
            infPaths = infPaths ∪ {P};
        else
            forall the j ∈ neighbors(i) do
                if (∑_{e∈P} x*_e + x*_{i,j} > |P|) ∧ (j ∉ P) then
                    Paths[j] = Paths[j] ∪ {P + {j}};
                    Q = Q ∪ {j}
                end
            end
        end
        setprocessed(P)
    end
end
Output: a set of infeasible paths infPaths, if such a path exists, and the empty
        set otherwise
```

### 4.3.3. NSGA-II

This section describes or implementation of NSGA-II for the CVRPB. The description of the single-objective GA for the DCVRP is omitted, since it is basically a simplified version of the NSGA-II[7] implementation.

**(1) Encoding of a solution.** We use a *giant tour representation* to encode a solution. Assume a route plan with $M$ different routes is given. Each route $R^m$ contains a subset of customer nodes and two copies of the depot node. Formally, $R^m = \left(0, i_1^m, i_2^m, \ldots, i_{p_i}^m, 0\right)$, $m \in \{1, 2, \ldots, M\}$, $p_i \in \mathbb{N}_0$ and $i_p^m \in V_0 \; \forall p \in \{1, 2, \ldots, p_i\}$. Then the corresponding giant tour uses $M + 1$ copies of the depot and is defined as

$$x = \left(0_1, i_1^1, \ldots, i_{p_1}^1, 0_2, i_1^2, \ldots, i_{p_2}^2, \ldots, 0_M, i_1^M, \ldots, i_{p_M}^M, 0_{M+1}\right).$$

**(2) Generation of the initial population.** For obtaining good initial solutions, we use a randomized savings algorithm Pasia et al.[96]. The traditional savings algorithm for the CVRP starts with routes each servicing only one customer. Iteratively, the partial routes are combined by selecting the two routes producing the maximum savings value $s(i,j) = c_{\{0,i\}} - c_{\{i,j\}} + c_{\{j,0\}}$, where only combinations $(i,j)$ leading to feasible routes are considered. This is repeated until no routes can be merged anymore. The randomized savings algorithm developed in Pasia et al.[96] maintains a candidate list $C$ of the feasible combinations with the $|C|$ best saving values, and selects one of them with equal probability.

**(3) Crossover.** We apply the crossover operator suggested by Prins[99]. Let two parent solutions $\pi_1$ and $\pi_2$ be given. First, we remove all trip delimiters from the encodings of the parent solutions. Then, an *order crossover* (OX) is performed on the parents to generate two children. OX constructs children $\gamma_1$ and $\gamma_2$ as follows: First, select two cutting points $i$ and $j$ in the first parent $\pi_1$. Then, copy the genes $(\pi_1[i], \ldots, \pi_1[j])$ into the first child $\gamma_1$, which gives $(\gamma_1[i], \ldots, \gamma_1[j])$. The remaining positions of $\gamma_1$ are then filled, starting at position $j+1$, by considering the genes in the order in which they appear in the second parent $\pi_2$ (if the end of the chromosome is reached, continue from the start). Duplications of genes are avoided by skipping elements that have already occurred. The second child is created analogously, changing the role of the parents. In order to partition the obtained string into different routes, a least-cost splitting procedure is used. This procedure is based on an exact shortest-path algorithm (Bellman's algorithm). First, the

[7] A general description is given in section 2.4.2.

minimum number of routes $\lambda$ is determined that is needed to split the given permutation of customers into routes such that each route fulfills the capacity restrictions and the distance restrictions. If this number is smaller than the number of available vehicles $K$, we split the given permutation into $K$, otherwise into $\lambda$ routes. The splitting is accomplished by a procedure similar to the algorithm that is used to determine the minimum number of required vehicles. For detailed information on these algorithms, see Chu et al.[26], Prins[99]. In the example in Figure 4.1, cutting points are chosen as $i = 3$ and $j = 5$.

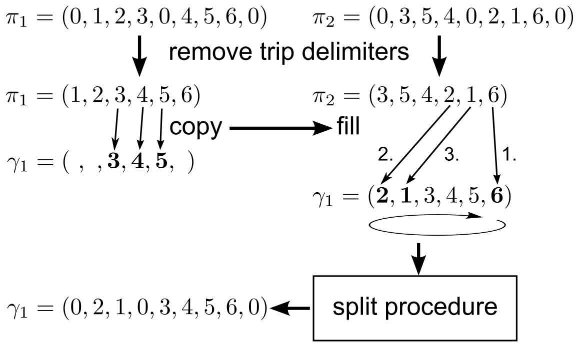



Figure 4.1.: OX Crossover and Split. 0 indicates trip delimiter; the cutting points are chosen as $i = 3$ and $j = 5$.

**(4) Mutation.** We use two simple mutation operators. Operator (i) exchanges two randomly chosen customers within one route by means of a 2-opt move. Operator (ii) exchanges a random number of customers between two different routes: Independently from each other, each customer is selected as an exchange candidate with a certain probability. For each exchange candidate, a random position in another route is chosen. Then the exchange candidate and the customer occupying the selected position in the other route change places. The choice between the application of operator 1 or 2 is performed randomly, governed by a probability which is a parameter of our implementation.

**(5) Constraint Handling.** We still have to describe how to handle the constraints of the multi-objective model during the execution of the NSGA-II, concerning (i) maximum number of vehicles (note that by the least-cost splitting procedure, a variable number of vehicles can result), and (ii) maximum capacity of the vehicles. Our operators applied to generate new solutions may create solutions that violate one or both of the mentioned constraints. Instead of relying on problem-specific repair functions to generate feasible solutions, we use the *constrained tournament method*[8] by Deb et al.[32], which is a kind of

[8] A description is given in section 2.4.2.

penalty function method implemented within the context of NSGA-II.

**(6) Elite-preserving procedure.** We used *controlled elitism* in our NSGA-II implementation (see Deb and Goel[30]). In contrast to the standard selection operator, where the new parent population $P_\mu$ is generated by successively copying all solutions of the $i$-th non-dominated front of $R_\mu$ to $P_\mu$, controlled elitism restricts the number of individuals that can be copied from the $i$-th front of $R_\mu$ to $P_\mu$. As suggested in Deb and Goel[30], a geometric distribution has been applied: The maximum number of allowed individuals in the $i$-th front in the new parent population $P_\mu$ of size $M_0$ is calculated as $M_i = M_0\,(1-r)\,r^{i-1}\,/\,(1-r^K)$ ($i = 1, 2, \ldots K$), where $0 < r < 1$. In combination with the constrained tournament method, not only feasible solutions are selected as parents, but also some solutions that are "slightly" infeasible, which helps to create diversity among the solutions of the Pareto set.

### 4.3.4. Implementation Details

**(1) General.** All algorithms were implemented in C++ and compiled by gcc version 4.3.2 with all optimization options enabled. We used the branch-and-cut framework of CPLEX 11.2. All tests were performed on a PC with 3.2 GHz.

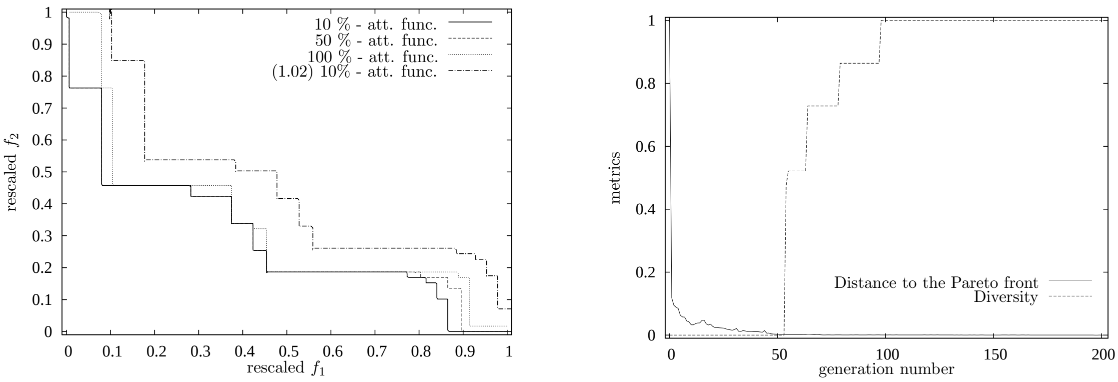
**(2) Caching.** As the calculation of the lower bound for the m-TSP is very time consuming, unnecessary recalculations have been avoided by implementing a caching feature. Each set of customers for which the lower bound for the m-TSP has been calculated was stored in a C++ map where a field representing the set of customers and the number $m$ of the vehicles was used as a key in order to quickly find the corresponding lower bound.

**(3) Separation strategy.** We followed the strategy by Lysgaard et al.[86] and treated the root node differently from the other nodes. With regard to the separation routines that address capacity constraints, we implemented the strategy in Lysgaard et al.[86]. For the separation routines that consider distance constraints, we used the following scheme. At the root node, separation stops if no new distance cut with a distance cut violation $> 0.001$ or infeasible path with violation $> 0.001$ for three subsequent LP re-optimizations was found. When separating distance constraints, the graph shrinking procedure is called first, then the connected components are checked for violated distance constraints and the separation routine for infeasible paths is called (at the root node the maximum number of infeasible paths is set to 1000, for the other nodes 10). If none of the aforementioned routines finds a violated constraint, the greedy randomized search is run. At non-root nodes, one run of the separation routines is performed, except of the greedy randomized search that is again only called if no violated constraints are found.

**(4) Branching.** If the separation routines do not find any violated constraints and the solution is not integer, we branch. Therefore, we use the strong branching feature of CPLEX.

**(5) Parameter choice for the metaheuristics.** For the heuristic algorithms NSGA-II and the single-objective genetic algorithm (SOGA), the following parameter settings obtained by computational experiments on selected test instances are used. We apply a crossover probability of 0.9 and a mutation probability of 0.1. For the mutation operators, the selection probability of each customer is set to 0.01, leading to mutations where only a small number of customers are exchanged. A fixed number of iterations (200) is used as the termination criterion. Population sizes are calculated dependent on the number of customers: population size $= 100 \lceil \text{number of customers}/10 \rceil$.

In Figure 4.2, the performance of the NSGA-II algorithm for a medium-sized selected test instance (A-n32-k5), including 32 customers and 5 vehicles, is shown.




(a) Attainment functions

(b) Running performance measures

Figure 4.2.: Performance measures for NSGA-II on test instance A-n32-k5 (cutoff factor $\infty$ for $f_1$).

In Figure 4.2(a), plots of the attainment functions[9] are shown for the selected test instance A-n32-k5 and 10 runs. The Figure shows that the worst-case front (100% attainment function, dotted curve) is very close to the best-case front (10% attainment function, solid curve; since we have 10 runs, this curve gives the border of the points that are dominated by at least one proposed solution). It can be concluded that for this instance, the

[9] A description is provided in Section 2.3.

algorithm is stable with respect to the random influence. In the figure, the values of $f_1$ and $f_2$ have been re-scaled to 0 for the minimum and 1 for the maximum value on the Pareto front; without re-scaling, the differences between the attainment functions are even smaller than they appear from the figure. To illustrate this, we have added the (1.02) 10% attainment function (dot-dashed curve) which represents the area that is dominated by the solutions of the best-case front, given that all objective function values of the solutions are multiplied by a factor 1.02. This shows that the worst-case front is always within a 2% gap to the best-case front.

Figure 4.2(b) plots two important runtime-related characteristics of multi-objective optimizers: (i) *distance* of points to the Pareto front, and (ii) *diversity evolution* of points[10].

Figure 4.2(a) and Figure 4.2(b) show that (i) for providing the exact branch-and-cut approach with good incumbents, it is sufficient to perform one run of NSGA-II, and that (ii) the chosen number of iterations is sufficiently high to guarantee stable results. Similar observations have been obtained for other test instances. The parameter choices for the SOGA have been assessed similarly.

## 4.4. Test Instances

To assess the performance of the algorithms, tests on a set of standard CVRP benchmark instances from the TSPLIB are used. For each of the test instances and each algorithm, we performed one run with different upper cutoff values for the distance objective function $f_1$ and two different maximum runtimes (4h, 8h). The upper cutoff values for $f_1$ are obtained by multiplying the minimal possible $f_1$ value by factors 1.05, 1.10, 1.15, 1.2 and $\infty$, respectively. This procedure has been chosen because in applications, the decision maker is usually not interested in solutions where the $f_1$ value (expressing transportation cost) is *considerably* higher (say, by 20% or more) than the best-possible value: the tradeoff to route balance is only of interest within a certain reasonable bandwidth of expenditure increments over the cheapest solution. This aspect is of special importance since, as it turns out, it is just the computation of the rightmost part of the Pareto front the which causes the largest computational effort. Thus, in applications, it makes sense to truncate the Pareto front at the right end by some pre-defined percentage of the optimal $f_1$ value. Of course, the relation between computational effort and width of the Pareto front window is of methodological interest; for this reason, we report on the results with the five factors (including $\infty$) indicated above. We selected 54 instances, covering problem sizes between 16 and 57 customers and 2 to 9 vehicles. As the locations of the customers are given as

[10] Descriptions of the used measures are given in Section 2.3.3.

coordinates in the plane,the distance between each pair of customers has been calculated as the Euclidean distance between the given points in the plane. This value has then been rounded to the nearest integer value, and finally an all-pairs shortest path algorithm has been run on the distances to ensure that the triangle inequalities are fulfilled.

## 4.5. Results

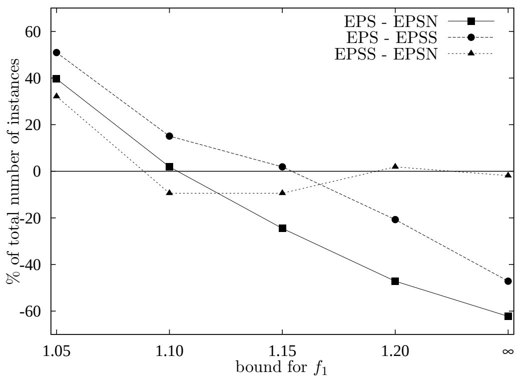
Table 4.1 shows the number of instances that each algorithm was able to solve within a given runtime limit. An instance is considered as solved if the algorithm is able to find all Pareto-optimal solutions within the given bound for $f_1$ and prove that there does not exist another solution within this bound. Results for two different maximum runtime limits (4h and 8h) are presented. In all tables, EPS denotes the pure $\varepsilon$-constraint method, EPSN denotes EPS + NSGA-II, and EPSS denotes EPS + SOGA.

| Bound: | 1.05 | | 1.10 | | 1.15 | | 1.20 | | $\infty$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| Runtime: | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| Total Solved | 37 | 39 | 28 | 28 | 20 | 21 | 15 | 18 | 9 | 10 |
| EPS | 36 | 38 | 28 | 28 | 20 | 20 | 15 | 17 | 8 | 9 |
| EPSN | 37 | 39 | 28 | 28 | 20 | 21 | 15 | 17 | 9 | 9 |
| EPSS | 37 | 38 | 28 | 28 | 20 | 20 | 15 | 15 | 9 | 10 |
| % of runtime (a) | 43 | 40 | 51 | 49 | 57 | 57 | 64 | 62 | 65 | 65 |
| % of runtime (b) | 74 | 67 | 72 | 70 | 71 | 71 | 73 | 70 | 73 | 72 |

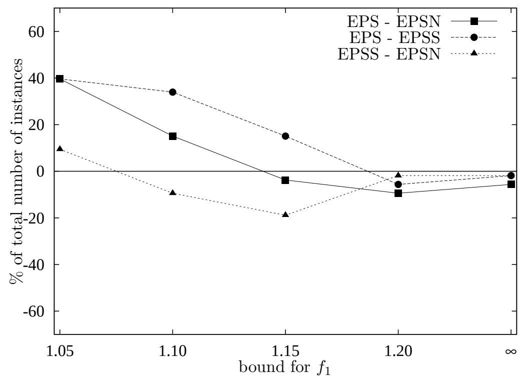
Table 4.1.: Number of solved instances, % of runtime: shows the average percentage of runtime that is used to prove that the Pareto set is complete (a) for all instances, (b) for not solved instances.

It is clear that as the bound for $f_1$ gets larger, the instances become harder to solve. Given tight bounds for $f_1$, about 68% of the instances were solved by each of the algorithms within 4h. This value rapidly decreases as the bound increases. An explanation for the high number of unsolved instances in the absence of a $f_1$ bound is that proving that there does not exist another feasible solution given a bound for $f_2$ (i.e., showing that a DCVRP with $D = f_2 - \Delta$, where $f_2$ is the value of the second objective function of the last found solution, is infeasible), is a hard problem that cannot be alleviated anymore by determining bounds from incumbent solutions. As a consequence, the search tree of the branch-and-cut algorithm quickly grows in this situation, as there is no upper bound available that allows to prune certain subtrees; the only case where a subtree can be discarded occurs if the LP relaxation at the current node is infeasible. As $\Delta$ has the value 1 in our implementation we are right at the border (in terms of $D$ of the DCVRP) where the instance is solvable for $(f_2)$ respectively infeasible for $D = f_2 - 1$. On average over all instances and algorithms, more than 40% of the provided runtime has been used to prove that the complete Pareto set has already been found. As seen from Table 4.1, this value increases as the bound for $f_1$ increases, but considering only the instances that were not solved by any algorithm, the value remains rather constant at about 72%. We take account of the effect that such a large amount of runtime is needed to prove completeness of the Pareto set by providing separate evaluations for "runtime until finding the front" and "runtime until proving completeness of the front" in the following comparisons of the algorithms.

For the comparisons, we define that an algorithm $A$ is better than an algorithm $B$, written as $A \lhd_b B$, if either algorithm $A$ is able to produce a larger number of Pareto-optimal solutions than algorithm $B$ within the given runtime limit, or algorithm $A$ is able to produce the same number of Pareto-optimal solutions within a shorter runtime than algorithm $B$. In Figures 4.3(a) and 4.3(b), we show the quotient $(N(A \lhd_b B) - N(B \lhd_b A))/N^{tot}$, where $N^{tot}$ and $N(event)$ denotes the total number of instances and the number of instances for which *event* holds, respectively. The quotient is plotted for different pairs of algorithms and in dependence of different bounds for $f_1$, based on a maximum runtime of 4h. Figures 2(a) and 2(b) differ by the exact interpretation of "producing solutions within a shorter runtime" in the definition of $A \lhd_b B$: In Figure 4.3(a), the runtime to identify the last solution is used, whereas in Figure 4.3(b), the runtime needed to prove that the complete Pareto set (within the given $f_1$ bound) has been found is used as the comparison criterion.




(a) Dominance based on runtime until last found solution



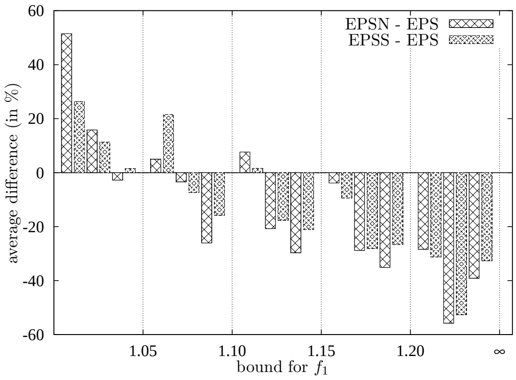

(b) Dominance based on runtime until proven complete

Figure 4.3.: Pairwise relative dominance count differences between the two hybrid algorithms and the adaptive $\varepsilon$-constraint method

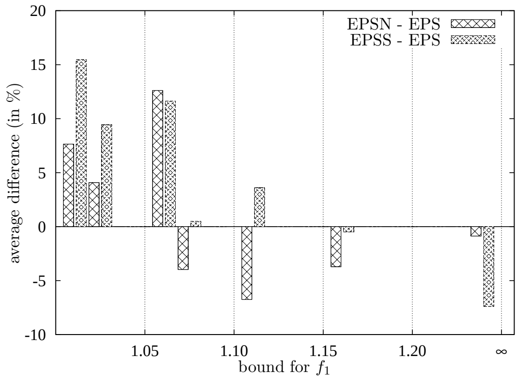
Figure 4.3(a) shows that for instances where the bound for $f_1$ is tight, the hybridization with the heuristics does not improve (or even worsens) the performance of the algorithm. Here, the time needed to heuristically generate incumbent candidates is not compensated by the achieved runtime reduction for the branch-and-cut algorithm. For larger bounds, one can see that it is useful to apply the heuristic algorithms to generate good incumbent candidates: For the three largest ones among the five considered bounds, the hybrid algorithms perform better than EPS. In a comparison between the two hybrid algorithms EPSN and EPSS, it is not clear which algorithm performs better; for the medium range of bounds, a slight superiority of EPSN can be recognized, but the effect is weak. Figure 4.3(b) demonstrates that the advantage of the hybrid algorithm diminishes if the runtime to prove completeness is used as the comparison criterion. This can be explained by the effect shown in Table 4.1 that proving completeness requires a large share of the runtime, together with the fact that for proving completeness, the hybrid algorithms possess no special advantage anymore since the heuristics can only provide an incumbent solution candidate if the problem is solvable.

To get a better insight into the possible runtime decrease achieved by the hybrid algorithms versus EPS, Figure 4.4(a) and Figure 4.4(b) show the runtime differences of the hybrid algorithms compared to EPS. For this analysis, we do not consider the five instances where the hybrid algorithms were able to identify a larger number of Pareto-optimal solutions than the EPS algorithm. Within each bound for $f_1$, the remaining instances were sorted in increasing order of the runtime needed by the EPS algorithm (representing the difficulty of the instances). Based on this sorted list, we created four groups (each of size 9 or 10) of instances and calculated the average runtime difference between the hybrid algorithms and the EPS algorithm. The figures show these differences for the three harder groups of instances, as in the first (easy) group, the EPS algorithm clearly outperforms the hybrid algorithms.

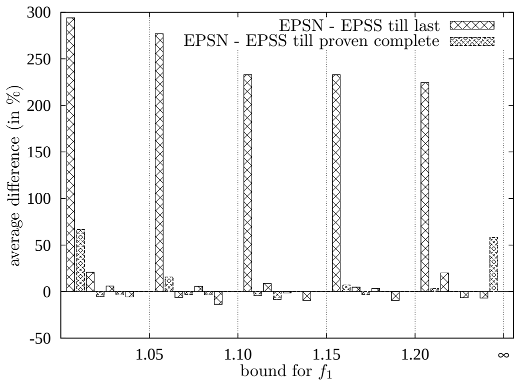
Figure 4.4(a) shows the performance of the hybrid algorithms compared to the EPS algorithm with respect to the runtime needed to identify the last found solution. As observed in Figure 4.3(a), the advantage of the hybrid algorithms increases as the bound for $f_1$ gets larger. It is important to notice that although in general EPS is the better algorithm when the bound for $f_1$ is tight, the performance of the hybrid algorithms compared to EPS tends to improve as the instances become more difficult to solve. In Figure 4.4(b), the influence of the comparably large runtime needed to prove completeness can be observed again. It is seen that with respect to the runtime until proven completeness, the advantage of the hybrid algorithms decreases. As before, for more difficult instances and $f_1$ bounds larger than 5%, the hybrid algorithms may have an advantage over the EPS algorithm, and




(a) Average difference in runtime until last found solution (in % of the runtime needed by EPS)




(b) Average difference in runtime until proven complete (in % of the runtime needed by EPS)



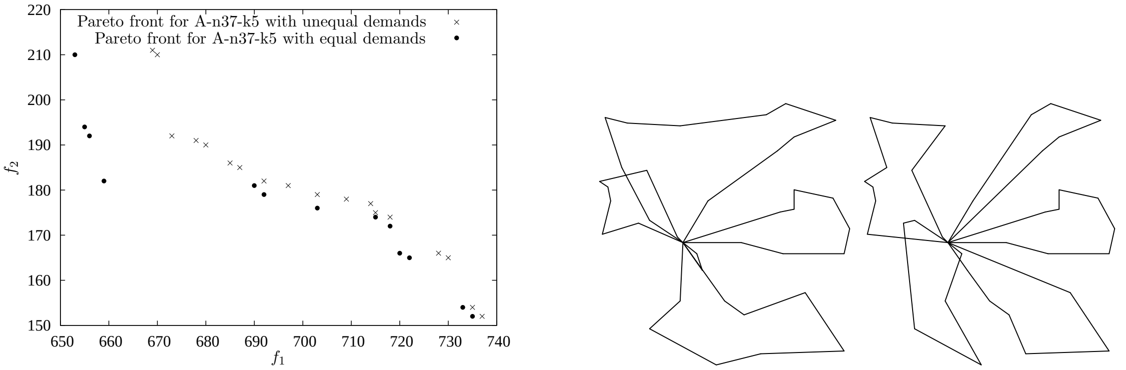

(c) Average difference in runtime (in % of the runtime needed by EPSN)

Figure 4.4.: Average runtime difference of the hybrid algorithms to the adaptive $\varepsilon$-constraint method

are at least not worse. Figure 4.4(c) illustrates the runtime differences between the hybrid algorithms EPSN and EPSS. In this figure, all four groups of instances are shown for each bound for $f_1$, as well as both comparisons (runtime until last found solution, runtime until proven complete). It can be observed that EPSS outperforms EPSN for easy instances, whereas for harder instances, no significant difference between the two algorithms can be noticed.

Finally, we present the complete Pareto front and the extreme solutions for a selected test case (A-n37-k5). In Figure 4.5(a), the Pareto front for the instance is shown (crosses). Comparing the leftmost and the rightmost point, we see that in this instance, it would be possible to decrease the length of the longest route by 28% at the expense of an increase in the total cost of about 10%. Such an analysis may be of interest if e.g. it is prefered that all vehicles return to the depot at the same time, but it is not known how to quantify this preference beforehand. Figure 4.5(b) presents the two extreme solutions of the given instance. The left figure shows the routes that have to be performed if only total cost is taken into account. As one can see, in this case, the routes are rather unbalanced (two long routes, one of medium length, one very short). The right figure shows the optimally balanced solution; here the lengths of all routes are almost the same.

It may be interesting to see what happens in the case where customer demands do not vary. The filled circles in Figure 4.5(a) show the Pareto front in the case where all demands have been set to the average of their values in the original instance. It can be observed that in both settings, the optimal solutions with respect to $f_1$ (total cost) have the nearly the same value of $f_2$ (maximum route length), but the optimal $f_1$ value is considerably reduced if the variance in the demand distribution is removed. This is intuitively plausible since the maximum route length is stronger influenced by the location of the customers than by their demands: a lower bound of the maximum route length (which can be tight in some instances) is given by twice the longest distance of a customer to the depot, independently of the demand. This consideration does not hold for the optimal total route length $f_1$, which is much more sensitive with respect to the vehicle capacity.




(a) Pareto front for test case A-n37-k5 (crosses). The circles show the changed Pareto front if customer demands are made equal by averaging.

(b) Extreme solutions for test case A-n37-k5 (left: solution with minimal $f_1$, right: solution with minimal $f_2$). Observe that both the left and the right solution contain five routes.

Figure 4.5.: Pareto front and extreme solutions for test case A-n37-k5.

## 4.6. Concluding Remarks

We developed exact hybrid algorithms for solving a bi-objective vehicle routing problem that does not only take minimization of total travel costs into account, but also considers the balance of routes as a second objective function. Our approach is based on the adaptive $\varepsilon$-constraint method for multi-objective combinatorial optimization problems and combines this method with two different metaheuristic algorithms (NSGA-II and a single-objective GA) in order to improve the performance. In the application of the adaptive $\varepsilon$-constraint method to our problem, an efficient exact algorithm is needed to solve the arising subproblems. In our implementation, the subproblems are DCVRPs, therefore we designed an efficient branch-and-cut algorithm for solving DCVRPs. A novel approach for treating the distance constraints by means of Held-Karp-type bounds was implemented.

We tested our proposed algorithms on a set of 54 CVRP benchmark instances from the TSPLIB. The computational experiments show that the implemented methods are capable of solving small to medium-sized instances to optimality. For harder instances, the hybrid algorithms perform distinctly better than the pure adaptive $\varepsilon$-constraint method, if the runtime to find the last Pareto-optimal solution is considered as the performance measure. This advantage decreases to some extent if the runtime to prove that the Pareto-set is complete is considered. Comparing the two different hybridization approaches, a sequential approach using NSGA-II and an interactive one using an ordinary GA, no significant differences between the two approaches could be observed.

Future research in several directions is possible; let us outline topics where such research will be particularly helpful. First, our experiments showed that efficiently proving the completeness of the Pareto set is a crucial issue for possible further improvements of the method. For the problem considered in this chapter, proving by a branch-and-cut approach that the last identified Pareto-optimal solution is the last existing Pareto-optimal solution is computationally very expensive, which suggests the application of other techniques for this purpose. Secondly, our branch-and-cut implementation could be substituted by a branch-and-price or a branch-and-cut-and-price algorithm. Third, the development of efficient separation algorithms for finding violated distance constraints might considerably improve the performance of the solution algorithms. Fourth, our approach could also be applied to different other variants of CVRP problem, e.g., the CVRP with time windows. Results can then be used as reference solution for evaluating different heuristic approaches. Fifth, other metaheuristics could be applied to generate incumbent candidates for the exact subproblem solver, these can either be multi-objective or single-objective algorithms. Sixth, from the modelling viewpoint, let us recall that in this chapter, we consider the fleet size $K$ as fixed. Of course, the user can solve the presented model for different candidate values of $K$ and compare the Pareto fronts, but a final picture cannot be obtained by simply superimposing the fronts, since acquiring or leasing an additional vehicle (and occupying one more driver) incurs additional expenses. Thus, an extension of the model making $K$ to a decision variable would be interesting. Finally, as in the project portfolio selection application, another typically encountered aspect in practical applications is uncertainty e.g. travel costs or the demands of customers are uncertain. This and the extension of the planning horizon, from one to several days, leads to a stochastic periodic vehicle problem, which we investigate at the moment. A brief description of the considered model, as well as a description of the proposed solution process is given in Section in the appendix.

# 5. Conclusion

In this thesis we presented different hybrid approaches for solving two different optimization problems in the field of multi-objective and stochastic multi-objective combinatorial optimization problems.

For the first application, the *Multi-objective Project Selection, Scheduling and Staffing with Learning* problem, we have developed a multi-objective model for project portfolio selection with respect to both economic and competence-oriented goals, and a bi-objective version of the model under uncertainty. The models also include different skill sets for the employees as well as learning and knowledge deprecation effects.

The stochastic version extends the deterministic model by including a third type of objectives that measures the robustness of portfolios in terms of expected surplus costs due to overtime work. We have implemented different solution approaches that rely on the decomposition of the model into two problems: (i) a discrete portfolio optimization problem as the master problem and (ii) a staffing problem, that is used to determine the assignment of available personnel to work packages as the subproblem. To solve the subproblem an approximation is presented by using a linear (mixed-integer, possibly stochastic) multi-objective program. In our implementation this linear subproblem is solved by a commercial solver (CPLEX). To solve the master problem we have implemented and investigated two metaheuristics based on the NSGA-II algorithm and the P-ACO algorithm. We assessed the performance of the algorithms on two different sets of test instances: (i) a set of randomly generated synthetic test cases of different size and type, and (ii) a real-world application delivered by the E-Commerce Competence Center Austria. Our computational studies showed that both hybrid algorithms provide reasonable solutions from a practical point of view.

To deal with the stochastic problem we implemented a procedure based on the APS (Adaptive Pareto Sampling) technique in combination with the aforementioned NSGA-II algorithm, and performed a computational study on a series of test instances derived from the real-world application indicated above. We compared the proposed technique to a complete enumeration approach with extensive simulation. Although our technique only consumed 1% of the runtime of the combined enumeration-simulation approach, the deviation of the solution quality was less than 1.6%. Concluding from these results, we anticipate that our technique will be well-suited also for solving test instances for which complete enumeration is not a feasible option anymore.

For the second application, the *Bi-objective Capacitated Vehicle Routing Problem with Route Balancing*, we developed and implemented exact hybrid algorithms for solving a bi-objective vehicle routing problem that does not only take minimization of total travel costs into account, but also considers the balance of routes as a second objective function. The presented approach is based on the adaptive $\varepsilon$-constraint method for multi-objective combinatorial optimization problems and combines this method with two different metaheuristic algorithms (NSGA-II and a single-objective GA) in order to improve the performance. In our application the adaptive $\varepsilon$-constraint method requires an efficient branch-and-cut algorithm for solving distance-constrained CVRPs, therfore we implemented a novel approach for treating the distance constraints by means of Held-Karp-type bounds. We performed computational experiments on a set of CVRP benchmark instances from the TSPLIB. These experiments show that the implemented methods are capable of solving small to medium-sized instances to optimality. For harder instances, the hybrid algorithms perform distinctly better than the pure adaptive $\varepsilon$-constraint method, if the runtime to find the last Pareto-optimal solution is considered as the performance measure. This advantage decreases to some extent if the runtime to prove that the Pareto-set is complete is considered. No significant difference between the different hybridization approaches could be observed. Our experiments showed that efficiently proving the completeness of the Pareto set is a crucial issue for possible further improvements of the method. For the considered problem, proving by a branch-and-cut approach that the last identified Pareto-optimal solution is the last existing Pareto-optimal solution is computationally very expensive, which suggests the application of other techniques for this purpose.

From a scientific point of view the hybrid approaches used in this thesis provide new methods to solve the considered problems. Especially in the first application where a linear problem is part of the considered optimization problem, the hybrid approach combining a heuristic procedure to solve the combinatorial part of the problem, and a LP solver to tackle the linear parts of the problem, provides much better results in terms of solution quality than a pure heuristic approach. In the second application where we want to determine the exact Pareto-front of a multi-objective optimization problem, hybrid approaches outperform pure exact methods in the case of "hard" problem instances.

# A. Work in Progress

## A.1. Problem Description

In this chapter, we present a brief description of an hybrid heuristic algorithm for solving a bi-objective stochastic periodic vehicle routing problem. The considered problem is a stochastic and bi-objective extension of the periodic vehicle routing problem with service choice (PVRP-SC) by Francis et al.[44]. Let us shortly recall the definition of the PVRP-SC. Its objective is to find optimal routes that are constructed for a period of time. In the classical periodic vehicle routing problem (PVRP) customers are visited a preset number of times over the period, the visit schedules for each customer are chosen from a fixed set of visit combinations. Each schedule represents a set of days on which a customer is visited. In the PVRP-SC the visit frequency of a customer is not a preset parameter of the model. For each customer a minimum number of visits per period is given, but higher frequencies are allowed. Francis et al.[44] describe the benefits of higher service frequencies in general as the customer's willingness to pay for more frequent service. In contrast to the inventory routing problem (IRP), in the PVRP-SC, the amount delivered to a customer is determined by the assigned schedule (the accumulated deterministic demand till the next visit). The IRP treats the amount to deliver as a separate decision from service frequency. The objective of the PVRP-SC is to find optimal routes for a fleet of $K$ identical vehicles (each with maximum capacity $Q$) for each day over a period of time that minimizes an objective of total travel costs minus service benefit, subject to operational constraints (i) the total demand of customers on each route is at most $Q$, (ii) the minimum service frequency for each customer is fulfilled, (iii) each customer is visited exactly once at the day a visit is required, (iv) each route at each day starts and ends at the depot and (v) no split deliveries are allowed.

As a generalization to the classical PVRP-SC we treat the demands $q_i$ of customers $i = 1, \ldots, n$ as *uncertain* values, and split the objective into two conflicting objectives: (i) the minimization of total travel costs as the first objective and (ii) the minimization of the total expected stockout of all customers as the second objective. This highlights the trade-off between travel costs and robustness of the assigned visit schedules. We also include that for each vehicle a maximum driving distance $D$ is given.

As in the application to project portfolio selection, for solving the resulting bi-objective stochastic optimization problem we apply Adaptive Pareto Sampling APS (cf. Section 2.4.1), combined with the Nondominated Sorting Genetic Algorithm II (NSGA-II)

This chapter is organized as follows: In Section A.2, we formulate the bi-objective stochastic extension of the PVRP-SC and introduce basic definitions. Section A.3 briefly explains the proposed solution techniques, i.e., the algorithms APS and NSGA-II as well as their interplay.

## A.2. Model Formulation

This section describes our model for the stochastic bi-objective PVRP-SC. It can be seen as an extension of the model used by Francis et al.[44] for the PVRP-SC. We assume that the travel time matrix and cost matrix coincide (this matrix is denoted by $C$) and that no service times are present. The elements of $C$ are supposed to fulfill the triangle inequality (i.e., the distance function is a metric). The stochastic bi-objective PVRP-SC with (i) minimization of the total cost and (ii) minimization of the total expected stockout can then be formulated as follows. The problem is defined on a directed graph $G = (V, A)$, where $V = \{0, 1, \ldots, n\}$ is the set of vertices, and $A = \{\{i, j\} : i, j \in V\}$ is the set of arcs. Index 0 denotes the depot, where a set $K$ of vehicles of capacity $Q$ and maximum allowable route length $D$ are located. The set of customers is given as $V_0 = V \setminus \{0\}$. The set $T = \{1, \ldots, t\}$ represents the set of days, where $t$ represents the length of the period. Each customer $i$ has an random nonnegative demand at each day $d \in T$ that is represented by a random vector $q_i(\omega) = [q_{i1}(\omega), \ldots, q_{it}(\omega)]$, where $\omega$ denotes the influence of randomness. In the following the average demand $\mathbb{E}(q_{id}(\omega))$ of a customer $i$ is the same for each day $d$; we denote this average demand by $\bar{q}_i$. (A generalization to varying average demands for different days is easily possible.) The minimum number of visits for each customer $i$ is denoted by $f_i$. By the matrix $C$, to each arc $(i, j) \in A$, a cost value $c_{ij}$ is associated, which can also be interpreted as the travel time or as the length of arc $(i, j)$. $S = \{1, \ldots, |S|\}$ denotes the set of all service schedules, where each $s \in S$ is a subset of $T$. Each $s$ is represented by a vector $a^s$ indexed by $d \in T$, where $a^s_d = 1$, if $d \in T$ is in schedule $s \in S$, otherwise $a^s_d = 0$. The variable $\gamma^s$ denotes the service frequency for schedule $s \in S$. In contrast to the traditional PVRP-SC formulation we do not use a single value $\beta^s$ (estimated as the maximum number of days between visits on schedule $s$) as the demand accumulation adjustment factor but use a vector $\beta^s$ indexed by $d \in T$. The elements of $\beta^s$ represent the numbers of days between two consecutive deliveries and can be calculated by the following equation.

$$\beta_d^s = \begin{cases} 0, & a_d^s = 0 \\ \min\left\{d' | d' > d, a_{d'}^s = 1\right\} - d, & \exists d' > d : a_{d'}^s = 1 \\ t - d + \min\left\{d' | a_{d'}^s = 1\right\}, & \text{otherwise} \end{cases} \tag{A.1}$$

E.g. if we consider a planning period of five days, then one possible schedule $s$ could be represented by $a^s = [0, 1, 0, 0, 1]$ or equivalently by $\beta^s = [0, 3, 0, 0, 2]$. To formulate the stochastic bi-objective PCVRP-SC as a stochastic mixed integer program (MIP), we define binary variables $y_i^s$ equal to 1 if and only if visit combination $s \in S$ is assigned to customer $i$, and $x_{ijk}^{sd}$ equal to 1 if and only if vehicle $k$ visits customer $j$ immediately after customer $i$ during day $d$ and a given schedule $s \in S$. Variables $z_i^d(\omega)$ represent the inventory of customer $i$ at the end of day $d$. We omit the constraint of the classical PVRP-SC that each customer needs to be visited by the same vehicle each time. The stochastic bi-objective PVRP-SC (SB-PVRP-SC) can be formulated as follows.

$$\min \ \left(f_1(x,y), f_2(y)\right) \tag{A.2}$$

$$f_1(x,y) = \sum_{d\in T}\sum_{k\in K}\sum_{s\in S}\sum_{(i,j)\in A} c_{ij} x_{ijk}^{sd}, \tag{A.3}$$

$$f_2(y) = -\,\mathbb{E}\left(\sum_{i\in V_0}\sum_{d\in T}\left[z_i^d(\omega)\right]^-\right), \tag{A.4}$$

$$z_i^d(\omega) = z_i^{d-1}(\omega) + \sum_{s\in S}\beta_d^s \bar{q}_i y_i^s - q_{id}(\omega) \qquad \forall i \in V_0, d \in T \setminus \left\{0, \min\left\{d' | a_{d'}^s y_i^s = 1\right\}\right\}, \tag{A.5}$$

$$z_i^d(\omega) = z_i^t(\omega) + \sum_{s\in S}\beta_d^s \bar{q}_i y_i^s - q_{id}(\omega) \qquad \forall i \in V_0, d \in \{0\} \setminus \min\left\{d' | a_{d'}^s y_i^s = 1\right\}, \tag{A.6}$$

$$z_i^d(\omega) = z_i^{init}(\omega) + \sum_{s\in S}\beta_d^s \bar{q}_i y_i^s - q_{id}(\omega) \qquad \forall i \in V_0, d = \min\left\{d' | a_{d'}^s y_i^s = 1\right\}, \tag{A.7}$$

$$\text{s.t.} \ \sum_{s\in S}\gamma^s y_i^s \geq f_i \qquad \forall i \in V_0, \tag{A.8}$$

$$\sum_{k\in K}\sum_{j\in V} x_{ijk}^{sd} = a_d^s y_i^s \qquad \forall i \in V_0, s \in S, d \in T, \tag{A.9}$$

$$\sum_{j\in V} x_{ijk}^{sd} = \sum_{j\in V} x_{jik}^{sd} \qquad \forall i \in V, s \in S, d \in T, k \in K, \tag{A.10}$$

$$\sum_{s\in S}\sum_{j\in V_0} x_{0jk}^{sd} \leq 1 \qquad \forall d \in T, k \in K, \tag{A.11}$$

$$\sum_{s\in S}\sum_{j\in V}\beta_d^s \bar{q}_i x_{ijk}^{sd} \leq Q \qquad \forall i \in V_0, d \in T, k \in K, \tag{A.12}$$

$$\sum_{s\in S}\sum_{(i,j)\in A} c_{ij}x_{ijk}^{sd} \le D \qquad \forall d \in T, k \in K, \tag{A.13}$$

$$\sum_{i\in V'}\sum_{j\in V'} x_{ijk}^{sd} \le |V'| - 1 \qquad \forall V' \subseteq V_0, V' \neq \emptyset, s \in S, d \in T, k \in K, \tag{A.14}$$

$$y_i^s \in \{0,1\} \qquad \forall i \in V_0, s \in S,$$

$$x_{ijk}^{sd} \in \{0,1\} \qquad \forall i,j \in V, s \in S, d \in T, k \in K,$$

$$z_i^d(\omega) \in \mathbb{R} \qquad \forall i \in V_0, d \in T,$$

$$z_i^{init}(\omega) \in \mathbb{R} \qquad \forall i \in V_0.$$

(A.3) is the classical objective of minimizing the total travel costs, (A.4) in combination with the inventory balance equations (A.5) - (A.7) form the second objective function: minimization of total expected stockout[1]. Constraint (A.8) defines that only schedules that fulfill the minimum visit frequency are allowed. (A.9) guarantees that each customer is visited exactly once on the days corresponding to the assigned visit combination. (A.10) impose that if a vehicle enters a node $i$ at day $d$ the vehicle also leaves the node at day $d$. Constraints (A.11) specify that each vehicle is used at most one a day. (A.12) and (A.13) are the constraints that limit the capacity and duration of each route. And (A.14) are the standard subtour elimination constraints. Using constraint (A.8) some $x_{ijk}^{sd}$ and $y_i^s$ can be fixed to value 0 in advance. If a schedule is not feasible with respect to constraint (A.8) for customer $i$, $y_i^s$ can be fixed to $y_i^s = 0$. The same can be done for variables $x_{ijk}^{sd}$, where schedule $s$ is not feasible with respect to constraint (A.8) for customer $i$ or $j$. Also variables $x_{ijk}^{sd}$ where the corresponding values $a_d^s = 0$ can be fixed to $x_{ijk}^{sd} = 0$.

## A.3. Solution Techniques

### A.3.1. General Approach

The proposed approach to solve SB-PVRP-SC problems is very similar to the method already successfully applied to the project portfolio selection problem of Chapter 3. In general the SB-PVRP-SC belongs to a computationally hard class of problems. It is immediately seen that already the deterministic, single objective special case obtained by removing the second objective $f_2$ (A.4) and the related inventory balance equations (A.5) - (A.7) is a generalization of the well known PVRP which is known to be $\mathcal{NP}$-hard. The presence of the $f_2$ term and the bi-objective situation further increase the complexity. For

[1] We are aware that $z_i^d(\omega)$ of the current week may be different than the week before. But we assume that our formulation provides a sufficient approximation.

most distributions of $q_{id}$, a direct evaluation of $f_2$ by numerical methods is costly or even impossible. For this reason, we resort to Monte-Carlo simulation to obtain an estimate of $f_2$ for each given $x$. Since we do not obtain exact evaluations of $f_2$ in this way, we apply APS in combination with a variant of the well known NSGA-II algorithm to solve the problem.

### A.3.2. NSGA-II

In this section we describe the components of the NSGA-II algorithm that are customized for the considered optimization problem.

**(1) Encoding of a solution.** Considering a discrete time period $T$ of $t$ days. Each customer is visited $u_i$ ($f_i \leq u_i \leq t$) times, but at most once per day. The total number of visits in $T$ is then the sum of all visits ($ns\,(T) = \sum_{i \in V_0} u_i$). Any solution for the PVRP-SC is a sequence of $ns\,(T)$ customers, divided into $t$ sublists. Each sublist represents the routes that have to be executed at a given day. To represent the the routes of a given day $d$ we use a *giant tour representation*. In general a giant tour represents a set of routes $R$ by combining the routes to one large tour that visits $|R|$ times the depot node (see, e.g.[68]). The repeated visits of the depot node in giant tour are represented by $|R|$ copies of the depot node. In the remaining work they are called *trip delimiters*. A solution is feasible if each customer $i$ appears at least $f_i$ times in $f_i$ distinct giant tours, according to one allowed schedule in $S$. All sub-paths of giant tours between two consecutive depot nodes need to represent feasible routes and the limited fleet size constraints must be fulfilled.

**(2) Generation of the initial population.** For obtaining good initial solutions, we randomly assign a feasible service schedule $s \in S$ that fulfills $\gamma_s \geq f_i$ to each customer $i \in V_0$. After the service schedules are fixed we know which customers need to be visited at day $d$ and the demands for each customer. To generate the routes for each day we use a randomized savings algorithm (Pasia et al.[95]). The traditional savings algorithm for the CVRP starts with routes each servicing only one customer. Iteratively, the partial routes are combined by selecting the two routes producing the maximum savings value $s\,(i,j) = c_{\{0,i\}} - c_{\{i,j\}} + c_{\{j,0\}}$, where only combinations $(i,j)$ leading to feasible routes are considered. This is repeated until no routes can be merged anymore. The randomized savings algorithm developed in (Pasia et al.[95]) maintains a candidate list $C$ of the feasible combinations with the $|C|$ best saving values, and selects one of them with equal probability.

**(3) Crossover.** We apply a variant of the crossover suggested by Lacomme et al.[77] adapted for the case that the service frequency for each customer is not fixed to a certain value but a minimum visit frequency is given. Let two parent solutions $\pi_1$ and $\pi_2$ be given. In the encoding of a solution, the routes driven by the vehicles at a certain day are represented by giant tours including trip delimiters. In general a *periodic linear order crossover* (PLOX) (Lacomme et al.[77]) is performed on the parents to generate two children. To perform the PLOX operator we first eliminate all trip delimiters from the giant tours, and combine the sequences of the single days to one long sequence, and define a target frequency $tf_i$ for each customer $i$. Then PLOX constructs children $\gamma_1$ and $\gamma_2$ as follows: First, select two cutting points $i$ and $j$ in the first parent $\pi_1$. Then, copy the customers $(\pi_1 [i], \ldots, \pi_1 [j])$ into the first child $\gamma_1$, while keeping their service days and order, and update the the target frequencies of the customers that are copied from $\pi_1$ as follows: $tf_i = min\,(f_i, \text{occurrence of } i \text{ in the copied sequence})$. The remaining positions of $\gamma_1$ are then filled, starting at the beginning of $\pi_2$, by considering the customers in the order they appear in $\pi_2$. For the customers that are not copied from $\pi_1$ we define the target frequency as $tf_i = min\,(f_i, \text{occurrence of } i \text{ in } \pi_2)$. PLOX only copies a customer from $\pi_2$ only if its target frequency is not yet satisfied and tries to keep the services days of a customer $i$ in $\pi_2$. Given that there are not enough visits for customer $i$ in $\gamma_1$ there are two possible cases for the current customer $i$ of $\pi_2$:
(i) $i$ is appended to the customers of the same day, if there is a compatible feasible service schedule available and $i$ is not already inserted at the current day.
(ii), Otherwise, $i$ is appended to the earliest day compatible with any feasible service schedule.
In the way we defined the target frequencies it is always guaranteed that the minimum service frequency of each customer $i$ is fulfilled and feasible service schedules exist. To decompose a coding without trip delimiters into giant tours with trip delimiters we use a least-cost splitting procedure for each day. The least-cost splitting procedure is an adaption of the procedures used in (Chu et al.[26], Lacomme et al.[77], Prins[99]). We adapted them to the case that the number of vehicles is not fixed and distance constraints are present.

**(4) Mutation.** We use one simple mutation operator. The operator exchanges a number of randomly chosen customers within one randomly chosen giant tour. Independently from each other, each customer is selected as an exchange candidate with a certain probability. For each exchange candidate, a random position is chosen. Then the exchange candidate and the customer occupying the selected position in the change places. The number of possible exchanges is controlled by a parameter that represents a selection probability.

**(5) Primary Local Search.** After each genetic operator we apply a local search procedure if the considered solution is feasible. For each day of the solution we apply a local search algorithm based on a *first improvement* strategy. In this algorithm we consider the well known 2-opt* and 2-opt neighborhoods and use *sequential search* procedures to find improving neighbors. A detailed description of sequential search algorithms as well as a comparison to traditional lexicographic search is given in (Irnich et al.[68]).

**(6) Constraint Handling.** As in the application to vehicle routing (Chapter 4) we use the constrained tournament method[2] by Deb et al.[32] to handle the following constraints: (i) maximum number of vehicles (note that by the least-cost splitting procedure, a variable number of vehicles can result), (ii) maximum capacity of the vehicles, and (iii) maximum allowed route distances.

**(7) Elite-preserving procedure.** Again we use controlled elitism in our NSGA-II implementation (see Deb and Goel[30] and Section 4.3.3).

### A.3.3. Importance Sampling

In our experiments, we assume the random variables $q_{id}(\omega)$ to be independent and modeled by poison distributions $\mathrm{Pois}(\bar{q}_i)$ where $\bar{q}_i$ is the average demand of customer $i$. To estimate objective function $f_2(y)$, a sample of $s$ scenarios $\omega_1, \ldots, \omega_s$ is drawn, where each scenario $\omega_\nu$ consists of a matrix $Q^{(\nu)} = \left[q_{11}^{(\nu)}, \ldots, q1t^{(\nu)}; \ldots; q_{n1}^{(\nu)}, \ldots, q_{nt}^{(\nu)}\right]$ of i.i.d. random numbers $q_{id}^{(\nu)}$ distributed according to $\mathrm{Pois}(\bar{q}_i)$ $(i = 1, \ldots, n, d = 1, \ldots, t)$. According to (2.7), the estimator $\tilde{f}_2(y)$ for $f_2(y)$ is given by

$$\tilde{f}_2(y) = \frac{1}{s} \sum_{\nu=1}^{s} f_2\left(x, Q^{(\nu)}\right) \tag{A.15}$$

where (cf. (A.4) and (A.5) - (A.7))

$$f_2(y, Q^{(\nu)}) = -\sum_{i \in V_0} \sum_{d \in T} \left[z_i^{d(\nu)}\right]^-, \tag{A.16}$$

$$z_i^{d(\nu)} = z_i^{d-1(\nu)} + \sum_{s \in S} \beta_d^s \bar{q}_i y_i^s - q_{id}^{(\nu)} \qquad \forall i \in V_0, d \in T \setminus \left\{0, \min\left\{d' | a_{d'}^s y_i^s = 1\right\}\right\}, \tag{A.17}$$

[2] A description is given in section 2.4.2.

$$z_i^{d(\nu)} = z_i^{t(\nu)} + \sum_{s\in S} \beta_d^s \bar{q}_i y_i^s - q_{id}^{(\nu)} \qquad \forall i \in V_0, d \in \{0\} \setminus \min\left\{d' | a_{d'}^s y_i^s = 1\right\}, \tag{A.18}$$

$$z_i^{d(\nu)} = z_i^{init(\nu)} \sum_{s\in S} \beta_d^s \bar{q}_i y_i^s - q_{id}^{(\nu)} \qquad \forall i \in V_0, d = \min\left\{d' | a_{d'}^s y_i^s = 1\right\}. \tag{A.19}$$

To reduce the variance of the estimator $\tilde{f}_2(y)$ without paying the cost of increasing sample size, we use importance sampling (IS) in our experiments (see, e.g., Rubinstein and Kroese[107]). In our case, for estimating $f_2(y)$, we are only interested in events where the inventory $z_i^{d(\nu)}$ of some customer at some day is less than zero: if this is not the case, the term $\left[z_i^{d(\nu)}\right]^-$ in (A.16) is zero. This suggests to shift the distribution $\text{Pois}(\bar{q}_i)$ of $q_{id}(\omega)$ to $\text{Pois}(\bar{q}_i{}^+)$ with some $\bar{q}_i{}^+$ satisfying $\bar{q}_i < \bar{q}_i{}^+$, such that the above-mentioned event occurs more frequently during sampling. The corresponding likelihood ratio is

$$\lambda^{(3)}\left(u; \bar{q}_i, \bar{q}_i{}^+\right) = \frac{\chi(u; \bar{q}_i)}{\chi(u; \bar{q}_i{}^+)} = \exp\left(-u \ln\left(\bar{q}_i{}^+\right) + \bar{q}_i{}^+ + u \ln(\bar{q}_i) - \bar{q}_i\right),$$

where $\chi(u; \bar{q}_i)$ denotes the probability density of the poison distribution $\text{Pois}(\bar{q}_i)$ in point $u$. Note that the distributions $\text{Pois}(\bar{q}_i)$ and $\text{Pois}(\bar{q}_i{}^+)$ have the same support. By the assumed independence of the random variables $q_{id}(\omega)$, we can multiply the likelihood ratios corresponding to the single variables $q_{id}(\omega)$ to obtain the overall weight. Thus, we can replace (A.15) - (A.16) by

$$\tilde{f}_2{}^{IS}(y) = \frac{1}{s} \sum_{\nu=1}^{s} f_2^{IS}\left(x, Q^{(\nu)}\right), \tag{A.20}$$

and

$$f_2^{IS}\left(y, Q^{(\nu)}\right) = \sum_{i\in V_0} \sum_{d\in T} \lambda\left(q_{id}^{(\nu)}; \bar{q}_i, \bar{q}_i{}^+, d\right) \left[z_i^{d(\nu)}\right]^-, \tag{A.21}$$

$$\lambda\left(q_{id}^{(\nu)}; \bar{q}_i, \bar{q}_i{}^+, d\right) = \begin{cases} \prod_{d'=\hat{d}}^{d} \lambda^{(3)}\left(q_{id'}^{(\nu)}; \cdot\right), & d \geq \hat{d} = \min\left\{d' | a_{d'}^s y_i^s = 1\right\} \\ \prod_{d'=\hat{d}}^{t} \lambda^{(3)}\left(q_{id'}^{(\nu)}; \cdot\right) \prod_{d'=1}^{d} \lambda^{(3)}\left(q_{id'}^{(\nu)}; \cdot\right), & d < \hat{d} = \min\left\{d' | a_{d'}^s y_i^s = 1\right\} \end{cases}, \tag{A.22}$$

$$(A.17) - (A.19),$$

where $q_{id}^{(\nu)}$ is now sampled from $\text{Pois}(\bar{q}_i{}^+)$ instead of $\text{Pois}(\bar{q}_i)$ $(i = 1, \ldots, n, d = 1, \ldots, t)$. To shift the distribution, a parameter $\alpha$ is used to determine $\bar{q}_i{}^+ = \alpha \bar{q}_i$ for each customer $i$.

## A.4. Preliminary Concluding Remarks

The model and the proposed solution method as well as some preliminary results have been presented at the Matheuristics 2010 conference in Vienna (June 2010). At the moment computational experiments to assess the performance of the proposed method and the model are still going on.
The preliminary experiments show that the parameter $\alpha$ influences the amount of variance reductions, and that the optimal value $\alpha_i^* = \alpha_i^* \left(\bar{q}_i, t\right)$ of $\alpha$ for a given customer $i$ depends in a rather complicated way on the parameters $\bar{q}_i$ and $t$ of the model, and there seems to be no chance to compute it in advance by means of some closed-form expression. By using a precomputed $\alpha_i^*$ variance reductions of about 60 % compared to standard sampling could be observed. Research is going on in several directions; let us outline topics that are investigated at the moment. First, we want to show how the performance (in terms of runtime) of the proposed method increases by using importance sampling instead of the trivial standard sampling approach. Second, a deeper understanding of the APS method should be obtained by using the running performance measures of Section 2.3.3 to investigate the influences of different update function of the sample sizes used in the solution proposal and solution evaluation stages of the APS algorithm. Third, the overall performance of the proposed algorithm will be assessed by using performance metrics for multi-objective optimizers described in Section 2.3, and a set of adapted standard test instances for the PVRP. Finally, we want to investigate the capabilities of the proposed model from the view point of a decision maker, especially we want to highlight the differences of solutions which could be obtained by changing the following parameters of the model: (i) changes in the minimum visit frequency $f_i$ of the customers (e.g. using fixed frequencies, no minimum frequency, etc.) and (ii) changes in the tightness of the capacity constraints.

# B. Additional Results

## B.1. Vehicle Routing

In Section B.1.1, the points of the Pareto sets of the considered test instances are shown. Section B.1.2 lists all results of our computational experiments. In section B.1.3, an aggregated view on the results of the computational experiments is given. This information is the basis for figures 4.4(a), 4.4(b), and 4.4(c) and the corresponding descriptions in Section 4.5.

### B.1.1. Pareto Optimal Solutions

| Testcase | Solutions |
|---|---|
| A-n32-k5 | (783,267) (784,254) (796,236) C (829,234) (844,229) (852,224) (857,220) CC (907,219) (915,218) (920,215) (924,209) CO |
| A-n33-k5 | (661,185) (671,169) (678,167) (684,163) C (705,161) (716,160) (717,158) C (728,157) COO |
| A-n33-k6 | (741,172) (754,170) (766,158) (769,155) C (781,154) (788,151) CO **(830,150)** OO |
| A-n34-k5 | (778,188) (783,185) (785,177) (804,175) (805,173) C (836,172) OOOO |
| A-n36-k5 | (799,226) (831,225) (833,224) C (841,222) (852,216) (865,215) (877,214) C (892,213) CCC |
| A-n37-k5 | (669,211) (670,210) (673,192) (678,191) (680,190) (685,186) (687,185) (692,182) (697,181) C (703,179) (709,178) (714,177) (715,175) (718,174) (728,166) (730,165) (735,154) C (737,152) CCC |
| A-n37-k6 | (949,250) (955,247) (969,239) (970,232) (985,228) (995,226) C (1001,224) (1007,223) (1016,218) (1017,216) (1034,214) C (1061,210) COO |
| A-n38-k5 | (730,184) (735,181) (758,178) (761,176) (762,172) C (770,170) (785,164) COOO |
| A-n39-k5 | (822,212) (857,200) (858,199) (862,198) C (893,197) OOOO |
| A-n39-k6 | (830,220) (833,209) (834,203) (842,202) (858,201) C (874,200) (898,199) (900,190) C (923,189) **(942,188)** OOO |
| A-n44-k6 | (936,247) (938,244) (940,241) (943,237) (950,235) (954,228) (956,219) (959,218) (964,215) (971,206) (975,203) C (986,202) C (1038,201) (1048,200) OOO |
| A-n45-k6 | (944,204) (969,203) (979,202) (983,197) (984,196) C **(1010,193)** OOOO |
| A-n45-k7 | (1145,229) (1146,221) (1147,220) (1160,219) (1161,217) (1164,214) (1167,212) (1177,209) OOOOO |
| A-n46-k7 | (913,197) (919,195) (927,194) (928,192) (946,188) (951,187) (955,186) C (962,184) (983,183) (989,182) C (1035,181) (1045,180) ccO |
| A-n48-k7 | (1073,206) (1080,205) OOOOO |
| A-n53-k7 | (1010,225) (1011,219) (1016,205) (1019,203) (1024,202) (1027,200) (1029,199) (1036,198) COOOO |
| A-n54-k7 | (1167,228) **(1170,227)** OOOOO |
| A-n55-k9 | (1072,176) (1075,171) (1099,170) C (1128,169) **(1148,166)** OOOO |

Table B.1.: Pareto-optimal solutions for the considered test instances, for each bound on $f_1$ a delimiter is used to indicate the border of the Pareto-set that lies within the given bound. C,c indicates that it was possible to prove that the set within the given bound is complete within 4h respectively 8h, O denotes that it is not known whether the set is complete or not. Bold fonts indicate solutions that could be found within 8h of runtime.

| Testcase | Solutions |
|---|---|
| B-n31-k5 | (672,189) CCCCC |
| B-n34-k5 | (788,212) (789,202) OOOOO |
| B-n35-k5 | (955,247) (989,245) (997,244) c **(1004,243)** OOOO |
| B-n38-k6 | (804,211) (806,210) (811,197) (814,195) (822,184) (827,183) (828,176) CC (912,174) CCC |
| B-n39-k5 | (549,196) CCOOO |
| B-n41-k6 | (829,187) (842,185) (865,170) (866,169) CCCCC |
| B-n43-k6 | (742,171) (754,170) (758,169) (759,168) (762,167) (763,166) (765,165) (769,162) (772,161) (775,160) OOOOO |
| B-n44-k7 | (909,212) (924,199) (927,171) (949,170) (951,168) C (985,167) CC (1078,166) CC |
| B-n45-k5 | (751,215) (752,213) (758,212) **(786,206) (787,205)** c **(792,201) (793,200)** OOOO |
| B-n45-k6 | (678,156) (679,154) **(710,146)** OOOOO |
| B-n50-k7 | (741,151) C (807,146) (812,144) (814,141) CCOO |
| B-n50-k8 | (1309,242) OOOOO |
| B-n51-k7 | (1032,215) **(1045,198) (1048,197)** OOOOO |
| B-n52-k7 | (747,214) (753,155) (760,153) (784,151) COOOO |
| B-n56-k7 | (707,202) (710,188) (713,187) (724,186) (741,183) C (743,182) CCCC |
| B-n57-k7 | (1153,202) OOOOO |
| B-n57-k9 | (1598,249) (1602,248) OOOOO |

Table B.2.: Pareto-optimal solutions for the considered test instances, for each bound on $f_1$ a delimiter is used to indicate the border of the Pareto-set that lies within the given bound. C,c indicates that it was possible to prove that the set within the given bound is complete within 4h respectively 8h, O denotes that it is not known whether the set is complete or not. Bold fonts indicate solutions that could be found within 8h of runtime.

| Testcase | Solutions |
|---|---|
| E-n22-k4 | (375,113) CC (414,112) (421,110) Ccc |
| E-n23-k3 | (569,289) (570,283) (595,280) C (600,260) (619,258) C (637,253) (653,247) CC (687,245) (688,242) O |
| E-n30-k3 | (534,216) (542,210) (544,209) (547,206) (549,196) (550,195) (551,192) **(560,191)** OOOOO |
| E-n30-k4 | (503,184) (511,177) (515,174) (521,172) C (548,170) C (558,165) C (589,164) CO |
| E-n33-k4 | (835,265) (839,262) (843,260) (846,259) (857,257) (860,256) (865,255) (869,254) C (882,251) (883,249) (888,247) (890,245) (913,244) COOO |
| E-n51-k5 | (521,117) (527,113) (528,112) (533,111) OOOOO |
| P-n16-k8 | (450,68) CCCCC |
| P-n19-k2 | (212,114) CCCCO |
| P-n20-k2 | (216,118) (218,112) CCCOO |
| P-n21-k2 | (211,117) (217,116) (220,112) CCCcO |
| P-n22-k2 | (216,117) (223,112) CCCcO |
| P-n22-k8 | (602,109) (613,108) (629,107) C (635,106) (654,105) (660,104) CCCC |
| P-n23-k8 | (529,90) C (563,89) CCCC |
| P-n40-k5 | (458,98) COOOO |
| P-n45-k5 | (510,117) (513,113) (514,111) (525,109) (526,108) COOOO |
| P-n50-k7 | (554,118) (555,116) (557,103) (558,101) (560,97) (563,93) (572,92) (573,91) (580,90) (581,87) COOOO |
| P-n55-k7 | (568,117) OOOOO |
| P-n55-k8 | (588,121) (589,105) (590,98) (593,97) (594,96) (597,90) (612,89) cOOOO |

Table B.3.: Pareto-optimal solutions for the considered test instances, for each bound on $f_1$ a delimiter is used to indicate the border of the Pareto-set that lies within the given bound. C,c indicates that it was possible to prove that the set within the given bound is complete within 4h respectively 8h, O denotes that it is not known whether the set is complete or not. Bold fonts indicate solutions that could be found within 8h of runtime.

### B.1.2. Runtimes

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | $\infty$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| An32k5 | E | 255; 435 | 255; 435 | 1349; 1405 | 1349; 1401 | 1597; 2444 | 1597; 2440 | 5957; 8808 | 5957; 8778 | 8735;14400 | 8735;28800 |
| | EN | 361; 500 | 361; 500 | 1053; 1369 | 1053; 1364 | 1059; 1926 | 1059; 1926 | 5589; 6309 | 5589; 6314 | 5603;14400 | 5603;28800 |
| | ES | 267; 504 | 267; 503 | 1152; 1466 | 1155; 1464 | 1157; 2136 | 1160; 2132 | 5097; 5467 | 5097; 5462 | 5101;14400 | 5101;28800 |
| An33k5 | E | 291; 404 | 291; 403 | 623; 699 | 623; 699 | 709; 848 | 709; 847 | 764;14400 | 764;28800 | 904;14400 | 904;28800 |
| | EN | 370; 450 | 369; 449 | 721; 761 | 721; 761 | 772; 874 | 771; 873 | 775;14400 | 774;28800 | 778;14400 | 777;28800 |
| | ES | 361; 496 | 360; 495 | 769; 875 | 768; 873 | 790; 939 | 790; 939 | 791;14400 | 791;28800 | 793;14400 | 793;28800 |
| An33k6 | E | 190; 272 | 190; 271 | 388; 1296 | 388; 1294 | 418;14400 | 419;28800 | 482;14400 | 0;28800 | 0;14400 | 0;28800 |
| | EN | 277; 311 | 276; 310 | 472; 4369 | 472; 4302 | 473;14400 | 473;28800 | 473;14400 | 19115;28800 | 474;14400 | 19306;28800 |
| | ES | 359; 453 | 357; 452 | 585; 2287 | 585; 2263 | 586;14400 | 586;28800 | 587;14400 | 28719;28800 | 588;14400 | 28719;28800 |
| An34k5 | E | 286; 467 | 286; 466 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| | EN | 339; 520 | 339; 520 | 7512;14400 | 7523;28800 | 7512;14400 | 7531;28800 | 7512;14400 | 7538;28800 | 7512;14400 | 7546;28800 |
| | ES | 395; 636 | 395; 635 | 0;14400 | 25117;28800 | 0;14400 | 25117;28800 | 0;14400 | 25117;28800 | 0;14400 | 25117;28800 |
| An36k5 | E | 1198; 1297 | 1198; 1298 | 10900;12515 | 10900;12529 | 11398;12442 | 11451;12442 | 11451;13781 | 11468;14562 | 12344;14221 | 12344;14223 |
| | EN | 919; 1039 | 921; 1041 | 6045;14400 | 6105;22135 | 6197; 6247 | 6230; 6250 | 6257; 6657 | 6349; 6608 | 6368;10238 | 6377;10296 |
| | ES | 786; 1006 | 785; 1005 | 12476;14400 | 12602;13747 | 12514;14400 | 12737;14436 | 12519;13689 | 12749;13487 | 12535;13534 | 12764;13466 |
| An37k5 | E | 877; 995 | 877; 996 | 2756; 3356 | 2756; 3350 | 3258; 3295 | 3258; 3280 | 3307; 6298 | 3307; 6301 | 4250;14400 | 4351;28800 |
| | EN | 921; 987 | 920; 987 | 2268; 3872 | 2513; 3858 | 2513; 2931 | 2888; 2928 | 2521; 4082 | 2537; 4083 | 2547;14400 | 3850;28800 |
| | ES | 1098; 1228 | 1098; 1228 | 2875; 4009 | 2875; 4001 | 3536; 4343 | 3535; 4327 | 3544; 7174 | 3543; 7185 | 3558;14400 | 3556;28800 |
| An37k6 | E | 2112; 2300 | 2112; 2300 | 3901; 4085 | 3901; 4086 | 5925;14400 | 5925;28800 | 12013;14400 | 12013;28800 | 13997;14400 | 13997;28800 |
| | EN | 2716; 2844 | 2713; 2841 | 4057; 4202 | 4055; 4201 | 6217;14400 | 6216;28800 | 6247;14400 | 6368;28800 | 6286;14400 | 6196;28800 |
| | ES | 2576; 2767 | 2574; 2765 | 3884; 4133 | 3886; 4136 | 5909;14400 | 5910;28800 | 5993;14400 | 5994;28800 | 6084;14400 | 6083;28800 |
| An38k5 | E | 570; 641 | 570; 633 | 9152;14400 | 9149;28800 | 9149;14400 | 9149;28800 | 10723;14400 | 10727;28800 | 10873;14400 | 10727;28800 |
| | EN | 448; 491 | 448; 490 | 9065;14400 | 9084;28800 | 9065;14400 | 9084;28800 | 9065;14400 | 9074;28800 | 9065;14400 | 9069;28800 |
| | ES | 380; 484 | 382; 486 | 10000;14400 | 9986;28800 | 10000;14400 | 9986;28800 | 10000;14400 | 9986;28800 | 10000;14400 | 9986;28800 |
| An39k5 | E | 758; 935 | 758; 934 | 3145; 3541 | 3214; 3546 | 4213;14400 | 4251;28800 | 5260;14400 | 5260;28800 | 5260;14400 | 5260;28800 |
| | EN | 858; 984 | 856; 983 | 2125; 2637 | 2125; 2681 | 2134;14400 | 2134;28800 | 2097;14400 | 2098;28800 | 2057;14400 | 2059;28800 |
| | ES | 820; 1028 | 819; 1027 | 2155; 2503 | 2145; 2494 | 2166;14400 | 2167;28800 | 2183;14400 | 2184;28800 | 2211;14400 | 2211;28800 |
| An39k6 | E | 471; 608 | 471; 608 | 7764;10453 | 7805;10462 | 7764;14400 | 19663;28800 | 7764;14400 | 19663;28800 | 9815;14400 | 19663;28800 |
| | EN | 592; 677 | 592; 678 | 8061;12273 | 8130;12276 | 8061;14400 | 18689;28800 | 8061;14400 | 18689;28800 | 8061;14400 | 18689;28800 |
| | ES | 546; 698 | 546; 698 | 9021;10279 | 9030;10337 | 9021;14400 | 16499;28800 | 9021;14400 | 16499;28800 | 9021;14400 | 16499;28800 |

Continued on next page

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | $\infty$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| An44k6 | E | 4292; 4508 | 4292; 4505 | 7109; 7314 | 7109; 7315 | 0;14400 | 14924;28800 | 0;14400 | 14924;28800 | 0;14400 | 14924;28800 |
| | EN | 5082; 5194 | 5082; 5176 | 5140; 5290 | 5140; 5283 | 13856;14400 | 13856;28800 | 13856;14400 | 13856;28800 | 13856;14400 | 13856;28800 |
| | ES | 4073; 4487 | 4073; 4517 | 6318; 6550 | 6318; 6565 | 12528;14400 | 12528;28800 | 12528;14400 | 12528;28800 | 12528;14400 | 12528;28800 |
| An45k6 | E | 0;10660 | 0;10662 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| | EN | 7260; 7302 | 7255; 7314 | 7248;14400 | 7247;28800 | 7254;14400 | 7250;28800 | 7256;14400 | 7253;28800 | 7248;14400 | 7242;28800 |
| | ES | 7289; 7609 | 7283; 7451 | 7285;14400 | 7278;28800 | 7286;14400 | 7286;28800 | 7283;14400 | 7258;28800 | 7285;14400 | 7278;28800 |
| An45k7 | E | 0;14400 | 23333;28800 | 0;14400 | 23930;28800 | 0;14400 | 24289;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| | EN | 13114;14400 | 13112;28800 | 13120;14400 | 13118;28800 | 13129;14400 | 13140;28800 | 13146;14400 | 13152;28800 | 13163;14400 | 13168;28800 |
| | ES | 0;14400 | 25874;28800 | 0;14400 | 25869;28800 | 0;14400 | 25862;28800 | 0;14400 | 25852;28800 | 0;14400 | 25836;28800 |
| An46k7 | E | 3201; 4142 | 3201; 4127 | 10433;10618 | 10433;24321 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| | EN | 2864; 3752 | 2864; 3748 | 8702; 8808 | 8702; 8810 | 11932;14400 | 11922;18192 | 11919;14400 | 11926;14069 | 11923;14400 | 11914;28800 |
| | ES | 3009; 3785 | 3009; 3788 | 6676;14400 | 6679; 6893 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| An48k7 | E | 306;14400 | 306;28800 | 884;14400 | 884;28800 | 884;14400 | 884;28800 | 1185;14400 | 1185;28800 | 1185;14400 | 1185;28800 |
| | EN | 654;14400 | 654;28800 | 654;14400 | 654;28800 | 654;14400 | 654;28800 | 654;14400 | 654;28800 | 654;14400 | 654;28800 |
| | ES | 433;14400 | 433;28800 | 433;14400 | 433;28800 | 433;14400 | 433;28800 | 462;14400 | 461;28800 | 462;14400 | 461;28800 |
| An53k7 | E | 0;14400 | 0;17088 | 0;14400 | 11609;28800 | 11609;14400 | 11609;28800 | 11609;14400 | 11632;28800 | 14108;14400 | 14108;28800 |
| | EN | 6450; 9127 | 6450; 9203 | 6450;14400 | 6450;28800 | 6450;14400 | 6450;28800 | 6450;14400 | 6450;28800 | 6450;14400 | 6450;28800 |
| | ES | 8819;11588 | 8819;11630 | 8819;14400 | 8819;28800 | 8819;14400 | 8819;28800 | 8819;14400 | 8819;28800 | 8819;14400 | 8819;28800 |
| An54k7 | E | 11356;14400 | 23865;28800 | 11802;14400 | 23865;28800 | 11912;14400 | 23865;28800 | 12064;14400 | 26112;28800 | 12064;14400 | 26112;28800 |
| | EN | 6174;14400 | 18429;28800 | 6203;14400 | 18519;28800 | 6174;14400 | 18429;28800 | 6203;14400 | 18519;28800 | 6174;14400 | 18429;28800 |
| | ES | 8190;14400 | 26426;28800 | 8230;14400 | 26558;28800 | 8190;14400 | 26426;28800 | 8230;14400 | 26558;28800 | 8190;14400 | 26426;28800 |
| An55k9 | E | 1257; 2174 | 1257; 2176 | 4381;14400 | 0;28800 | 4580;14400 | 0;28800 | 4620;14400 | 22780;28800 | 5643;14400 | 25757;28800 |
| | EN | 1625; 2787 | 1625; 2785 | 4390;14400 | 21813;28800 | 4410;14400 | 21792;28800 | 4390;14400 | 21802;28800 | 4410;14400 | 21812;28800 |
| | ES | 1618; 2902 | 1618; 2902 | 3225;14400 | 17447;28800 | 3241;14400 | 17534;28800 | 3225;14400 | 17447;28800 | 3241;14400 | 17534;28800 |
| Bn31k5 | E | 1; 1 | 1; 1 | 1; 1 | 1; 1 | 1; 12 | 1; 12 | 1; 11 | 1; 11 | 1; 13 | 1; 368 |
| | EN | 47; 48 | 47; 47 | 47; 47 | 47; 47 | 47; 47 | 47; 47 | 47; 48 | 47; 47 | 47; 47 | 47; 47 |
| | ES | 7; 16 | 7; 16 | 7; 22 | 7; 22 | 7; 29 | 7; 28 | 7; 27 | 7; 27 | 7; 30 | 7; 35 |
| Bn34k5 | E | 213;14400 | 213;28800 | 233;14400 | 233;28800 | 244;14400 | 244;28800 | 245;14400 | 245;28800 | 250;14400 | 250;28800 |
| | EN | 197;14400 | 197;28800 | 197;14400 | 197;28800 | 197;14400 | 197;28800 | 197;14400 | 197;28800 | 197;14400 | 197;28800 |
| | ES | 225;14400 | 225;28800 | 226;14400 | 226;28800 | 225;14400 | 225;28800 | 226;14400 | 226;28800 | 225;14400 | 225;28800 |
| Bn35k5 | E | 3397;14400 | 3397;28800 | 8600;14400 | 8600;28800 | 8636;14400 | 21435;28800 | 8636;14400 | 21435;28800 | 8636;14400 | 21435;28800 |
| | EN | 2419;14400 | 2542;28800 | 2430;14400 | 2554;28800 | 2419;14400 | 14797;28800 | 2430;14400 | 14806;28800 | 2419;14400 | 14804;28800 |

Continued on next page

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | ∞ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| | ES | 3653;14400 | 3653;28800 | 3671;14400 | 3671;28800 | 3653;14400 | 23050;28800 | 3671;14400 | 23040;28800 | 3653;14400 | 23048;28800 |
| Bn38k6 | E | 670; 757 | 670; 755 | 736; 849 | 736; 848 | 1007; 1062 | 1007; 1061 | 1057;11660 | 1057;11652 | 11656;11661 | 11656;11666 |
| | EN | 747; 776 | 747; 775 | 750; 778 | 750; 777 | 991; 1032 | 991; 1032 | 995; 1541 | 995; 1542 | 991; 2214 | 991; 2210 |
| | ES | 695; 736 | 695; 737 | 698; 956 | 698; 955 | 987; 991 | 987; 1019 | 991; 1304 | 991; 1300 | 987; 1572 | 987; 1685 |
| Bn39k5 | E | 1; 2 | 1; 2 | 1; 96 | 1; 95 | 1;14400 | 1;28800 | 1;14400 | 1;28800 | 1;14400 | 1;28800 |
| | EN | 98; 112 | 98; 112 | 98; 138 | 98; 138 | 98;14400 | 98;28800 | 98;14400 | 98;28800 | 98;14400 | 98;28800 |
| | ES | 14; 33 | 14; 33 | 14; 133 | 14; 133 | 14;14400 | 14;28800 | 14;14400 | 14;28800 | 14;14400 | 14;28800 |
| Bn41k6 | E | 437; 519 | 437; 518 | 438; 536 | 438; 536 | 456; 570 | 456; 569 | 506; 542 | 506; 544 | 519; 611 | 519; 613 |
| | EN | 515; 585 | 515; 586 | 517; 593 | 517; 593 | 515; 575 | 515; 575 | 517; 562 | 517; 561 | 515; 630 | 515; 631 |
| | ES | 505; 543 | 505; 542 | 507; 651 | 507; 650 | 505; 633 | 505; 633 | 507; 683 | 507; 681 | 505; 640 | 505; 651 |
| Bn43k6 | E | 0;14400 | 27796;28800 | 0;14400 | 27806;28800 | 0;14400 | 27945;28800 | 0;14400 | 27806;28800 | 0;14400 | 27945;28800 |
| | EN | 13137;14400 | 13137;28800 | 13202;14400 | 13202;28800 | 13137;14400 | 13137;28800 | 13202;14400 | 13202;28800 | 13137;14400 | 13137;28800 |
| | ES | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| Bn44k7 | E | 681; 776 | 681; 776 | 790; 880 | 790; 882 | 859; 963 | 859; 962 | 1072; 1075 | 1072; 1074 | 1286; 1289 | 1286; 1328 |
| | EN | 876; 943 | 876; 943 | 952; 977 | 952; 978 | 956; 996 | 956; 996 | 1068; 1075 | 1068; 1075 | 1072; 1087 | 1072; 1087 |
| | ES | 707; 835 | 707; 835 | 997; 1342 | 997; 1338 | 1001; 1180 | 1001; 1183 | 1224; 1237 | 1224; 1238 | 1230; 1273 | 1230; 1237 |
| Bn45k5 | E | 300;14400 | 0;28800 | 338;14400 | 0;28800 | 712;14400 | 0;28800 | 1024;14400 | 0;28800 | 2293;14400 | 0;28800 |
| | EN | 473;14400 | 473;18818 | 474;14400 | 474;28800 | 473;14400 | 473;28800 | 474;14400 | 23330;28800 | 473;14400 | 23446;28800 |
| | ES | 392;14400 | 0;28800 | 393;14400 | 0;28800 | 392;14400 | 0;28800 | 393;14400 | 0;28800 | 392;14400 | 0;28800 |
| Bn45k6 | E | 1647;14400 | 28286;28800 | 2601;14400 | 28286;28800 | 2697;14400 | 28286;28800 | 3314;14400 | 28286;28800 | 10528;14400 | 28286;28800 |
| | EN | 3026;14400 | 25828;28800 | 3034;14400 | 25950;28800 | 3037;14400 | 26052;28800 | 3038;14400 | 26024;28800 | 3046;14400 | 26041;28800 |
| | ES | 3025;14400 | 25819;28800 | 3029;14400 | 25760;28800 | 3026;14400 | 25712;28800 | 3019;14400 | 25649;28800 | 3006;14400 | 25662;28800 |
| Bn50k7 | E | 1; 7 | 1; 7 | 9168;12834 | 9168;12862 | 11071;12136 | 11071;12148 | 11071;14400 | 11071;28800 | 12699;14400 | 12699;28800 |
| | EN | 43; 92 | 43; 92 | 6062; 6963 | 6062; 6966 | 6091;14400 | 6091;15910 | 6062;14400 | 6062;28800 | 6091;14400 | 6091;28800 |
| | ES | 32; 90 | 32; 90 | 6210; 7078 | 6210; 7073 | 6241;11189 | 6241;11165 | 6210;14400 | 6210;28800 | 6241;14400 | 6241;28800 |
| Bn50k8 | E | 9821;14400 | 9827;28800 | 9820;14400 | 9826;28800 | 9822;14400 | 9819;28800 | 9811;14400 | 9815;28800 | 9821;14400 | 9828;28800 |
| | EN | 9783;14400 | 9784;28800 | 9783;14400 | 9792;28800 | 9788;14400 | 9797;28800 | 9788;14400 | 9791;28800 | 9799;14400 | 9794;28800 |
| | ES | 9723;14400 | 9713;28800 | 9712;14400 | 9707;28800 | 9706;14400 | 9699;28800 | 9705;14400 | 9711;28800 | 9706;14400 | 9712;28800 |
| Bn51k7 | E | 12;14400 | 26017;28800 | 19;14400 | 26017;28800 | 19;14400 | 26017;28800 | 19;14400 | 26017;28800 | 20;14400 | 26017;28800 |
| | EN | 181;14400 | 15351;28800 | 181;14400 | 15351;28800 | 181;14400 | 15341;28800 | 181;14400 | 15351;28800 | 181;14400 | 15365;28800 |
| | ES | 40;14400 | 0;28800 | 40;14400 | 0;28800 | 40;14400 | 0;28800 | 40;14400 | 0;28800 | 40;14400 | 0;28800 |
| Bn52k7 | E | 1416; 1850 | 1416; 1850 | 1416;14400 | 1416;28800 | 2475;14400 | 2475;28800 | 4096;14400 | 4096;28800 | 4096;14400 | 4096;28800 |

Continued on next page

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | ∞ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| | EN | 1194; 1517 | 1194; 1519 | 1199;14400 | 1199;28800 | 1194;14400 | 1194;28800 | 1199;14400 | 1199;28800 | 1194;14400 | 1194;28800 |
| | ES | 1191; 1739 | 1191; 1744 | 1196;14400 | 1196;28800 | 1191;14400 | 1191;28800 | 1196;14400 | 1196;28800 | 1191;14400 | 1191;28800 |
| Bn56k7 | E | 1138; 1393 | 1138; 1394 | 1145; 1513 | 1145; 1506 | 1405; 1442 | 1405; 1440 | 1439; 1508 | 1439; 1503 | 1510; 1608 | 1510; 1607 |
| | EN | 1056; 1205 | 1056; 1204 | 1280; 1364 | 1280; 1364 | 1285; 1358 | 1285; 1356 | 1280; 1365 | 1280; 1362 | 1285; 1358 | 1285; 1352 |
| | ES | 1457; 2024 | 1457; 2022 | 1599; 1710 | 1599; 1710 | 1606; 1710 | 1606; 1705 | 1599; 1705 | 1599; 1702 | 1606; 1798 | 1606; 1799 |
| Bn57k7 | E | 105;14400 | 105;28800 | 111;14400 | 111;28800 | 122;14400 | 122;28800 | 128;14400 | 128;28800 | 1648;14400 | 1648;28800 |
| | EN | 343;14400 | 343;28800 | 343;14400 | 343;28800 | 343;14400 | 343;28800 | 343;14400 | 343;28800 | 343;14400 | 343;28800 |
| | ES | 135;14400 | 135;28800 | 135;14400 | 135;28800 | 135;14400 | 135;28800 | 135;14400 | 135;28800 | 135;14400 | 135;28800 |
| Bn57k9 | E | 3926;14400 | 3942;28800 | 3944;14400 | 3954;28800 | 3936;14400 | 3949;28800 | 3938;14400 | 3951;28800 | 3938;14400 | 3936;28800 |
| | EN | 3360;14400 | 3348;28800 | 3335;14400 | 3344;28800 | 3332;14400 | 3322;28800 | 3337;14400 | 3324;28800 | 3316;14400 | 3320;28800 |
| | ES | 3742;14400 | 3733;28800 | 3725;14400 | 3738;28800 | 3731;14400 | 3718;28800 | 3722;14400 | 3709;28800 | 3696;14400 | 3704;28800 |
| En22k4 | E | 1; 2 | 1; 2 | 1; 14 | 1; 14 | 47; 65 | 47; 65 | 79; 249 | 79; 249 | 104;14400 | 104;28800 |
| | EN | 47; 48 | 47; 48 | 47; 48 | 47; 48 | 77; 78 | 77; 79 | 77; 124 | 77; 124 | 77;14400 | 77;28800 |
| | ES | 8; 20 | 8; 20 | 8; 35 | 8; 35 | 53; 88 | 53; 87 | 53; 148 | 53; 150 | 53;14400 | 53;14413 |
| En23k3 | E | 13; 15 | 13; 15 | 102; 160 | 102; 160 | 241; 296 | 241; 296 | 409; 3664 | 409; 3664 | 0;14400 | 0;28800 |
| | EN | 50; 55 | 50; 55 | 138; 181 | 138; 180 | 272; 311 | 272; 310 | 273; 2275 | 273; 2282 | 8088;14400 | 8086;28800 |
| | ES | 40; 50 | 40; 49 | 156; 224 | 156; 223 | 305; 371 | 305; 370 | 306; 1646 | 306; 1646 | 10396;14400 | 10396;28800 |
| En30k3 | E | 1954;14400 | 28210;28800 | 1954;14400 | 28210;28800 | 3903;14400 | 28210;28800 | 5222;14400 | 28210;28800 | 7929;14400 | 28210;28800 |
| | EN | 2601;14400 | 18996;28800 | 2613;14400 | 19090;28800 | 2601;14400 | 18996;28800 | 2613;14400 | 19090;28800 | 2601;14400 | 18996;28800 |
| | ES | 2358;14400 | 24699;28800 | 2369;14400 | 24822;28800 | 2358;14400 | 24699;28800 | 2369;14400 | 24822;28800 | 2358;14400 | 24699;28800 |
| En30k4 | E | 67; 117 | 67; 117 | 212; 267 | 212; 267 | 250; 327 | 250; 326 | 504; 1379 | 504; 1379 | 930;14400 | 930;28800 |
| | EN | 107; 136 | 107; 135 | 176; 213 | 176; 213 | 244; 294 | 244; 294 | 422; 1070 | 422; 1070 | 423;14400 | 423;28800 |
| | ES | 112; 172 | 112; 171 | 233; 316 | 233; 314 | 267; 359 | 267; 359 | 403; 1168 | 403; 1168 | 405;14400 | 405;28800 |
| En33k4 | E | 743; 810 | 743; 810 | 2814; 5484 | 2814; 5492 | 2814;14400 | 2814;28800 | 2814;14400 | 2814;28800 | 4241;14400 | 4241;28800 |
| | EN | 826; 853 | 826; 852 | 2672; 5207 | 2669; 5181 | 2662;14400 | 2665;28800 | 2677;14400 | 2665;28800 | 2670;14400 | 2679;28800 |
| | ES | 924; 1005 | 924; 1004 | 2736; 5332 | 2741; 5333 | 2747;14400 | 2733;28800 | 2744;14400 | 2732;28800 | 2723;14400 | 2724;28800 |
| En51k5 | E | 2324;14400 | 2324;28800 | 5348;14400 | 5348;28800 | 7527;14400 | 7527;28800 | 7527;14400 | 7527;28800 | 7527;14400 | 7527;28800 |
| | EN | 1176;14400 | 1177;28800 | 1172;14400 | 1169;28800 | 1168;14400 | 1167;28800 | 1165;14400 | 1167;28800 | 1168;14400 | 1164;28800 |
| | ES | 2074;14400 | 2072;28800 | 2062;14400 | 2052;28800 | 2050;14400 | 2040;28800 | 2038;14400 | 2028;28800 | 2021;14400 | 2026;28800 |
| Pn16k8 | E | 73; 76 | 73; 76 | 74; 76 | 74; 76 | 76; 76 | 76; 76 | 76; 75 | 76; 76 | 78; 76 | 80; 76 |
| | EN | 99; 105 | 99; 105 | 99; 105 | 99; 105 | 99; 104 | 99; 105 | 99; 105 | 99; 104 | 99; 104 | 99; 104 |
| | ES | 81; 93 | 81; 93 | 81; 93 | 81; 94 | 81; 94 | 81; 94 | 81; 93 | 81; 94 | 81; 94 | 81; 94 |

Continued on next page

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | ∞ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| Pn19k2 | E | 2; 3 | 2; 3 | 2; 3 | 2; 3 | 2; 241 | 2; 240 | 2;13445 | 2;13434 | 2;14400 | 2;28800 |
| | EN | 15; 16 | 15; 16 | 15; 16 | 15; 16 | 15; 221 | 15; 221 | 15;14400 | 15;20995 | 15;14400 | 15;28800 |
| | ES | 2; 8 | 2; 8 | 2; 8 | 2; 8 | 2; 212 | 2; 212 | 2;12736 | 2;12701 | 2;14400 | 2;28800 |
| Pn20k2 | E | 1; 2 | 1; 2 | 1; 47 | 1; 46 | 1; 4063 | 1; 4056 | 1;14400 | 1;28800 | 1;14400 | 1;28800 |
| | EN | 17; 18 | 17; 18 | 17; 58 | 17; 57 | 17; 3956 | 17; 3960 | 17;14400 | 17;28800 | 17;14400 | 17;28800 |
| | ES | 7; 16 | 7; 15 | 7; 58 | 7; 58 | 7; 4118 | 7; 4112 | 7;14400 | 7;28800 | 7;14400 | 7;28800 |
| Pn21k2 | E | 1; 2 | 1; 2 | 1; 5 | 1; 5 | 1; 116 | 1; 116 | 1;14400 | 1;14400 | 1;14400 | 1;28800 |
| | EN | 17; 20 | 17; 20 | 17; 18 | 17; 20 | 17; 118 | 17; 118 | 17;14400 | 17;24244 | 17;14400 | 17;28800 |
| | ES | 8; 13 | 8; 13 | 8; 26 | 8; 25 | 8; 125 | 8; 124 | 8;14400 | 8;28800 | 8;14400 | 8;28800 |
| Pn22k2 | E | 1; 2 | 1; 2 | 1; 13 | 1; 12 | 1; 1325 | 1; 1324 | 1;14400 | 1;14400 | 1;14400 | 1;28800 |
| | EN | 35; 46 | 35; 46 | 35; 46 | 35; 46 | 35; 1204 | 35; 1206 | 35;14400 | 35;28800 | 35;14400 | 35;28800 |
| | ES | 13; 23 | 13; 22 | 13; 31 | 13; 31 | 13; 1273 | 13; 1272 | 13;14400 | 13;28800 | 13;14400 | 13;28800 |
| Pn22k8 | E | 255; 382 | 255; 382 | 530; 601 | 530; 601 | 601; 784 | 601; 783 | 670; 879 | 670; 879 | 706;14400 | 706; 1505 |
| | EN | 293; 399 | 293; 399 | 587; 636 | 587; 636 | 589; 732 | 589; 732 | 587; 1255 | 587; 1266 | 589;12885 | 589; 1639 |
| | ES | 284; 422 | 284; 421 | 597; 676 | 597; 676 | 599; 781 | 599; 780 | 597; 824 | 597; 823 | 599; 1614 | 599; 1734 |
| Pn23k8 | E | 2; 164 | 2; 163 | 237; 724 | 237; 725 | 529; 532 | 529; 532 | 721; 2399 | 721; 2404 | 2397; 2410 | 2397; 2408 |
| | EN | 43; 191 | 43; 190 | 183; 337 | 183; 336 | 183; 299 | 183; 299 | 183; 2332 | 183; 2342 | 183; 2352 | 183; 2356 |
| | ES | 10; 167 | 10; 167 | 528; 754 | 528; 754 | 530; 540 | 530; 540 | 528; 2381 | 528; 2384 | 530; 2396 | 530; 2393 |
| Pn40k5 | E | 1; 54 | 1; 54 | 1;14400 | 1;28800 | 1;14400 | 1;28800 | 1;14400 | 1;28800 | 1;14400 | 1;28800 |
| | EN | 81; 97 | 81; 97 | 81;14400 | 81;28800 | 81;14400 | 81;28800 | 81;14400 | 81;28800 | 81;14400 | 81;28800 |
| | ES | 18; 95 | 18; 95 | 18;14400 | 18;28800 | 18;14400 | 18;28800 | 18;14400 | 18;28800 | 18;14400 | 18;28800 |
| Pn45k5 | E | 1005; 4798 | 1005; 4825 | 1541;14400 | 1541;28800 | 6115;14400 | 6115;28800 | 6115;14400 | 6115;28800 | 6115;14400 | 6115;28800 |
| | EN | 1758; 5625 | 1758; 5627 | 1766;14400 | 1766;28800 | 1758;14400 | 1758;28800 | 1766;14400 | 1766;28800 | 1758;14400 | 1758;28800 |
| | ES | 1143; 5007 | 1143; 5019 | 1148;14400 | 1148;28800 | 1143;14400 | 1143;28800 | 1148;14400 | 1148;28800 | 1143;14400 | 1143;28800 |
| Pn50k7 | E | 1925; 2026 | 1925; 2031 | 2611;14400 | 2611;28800 | 3994;14400 | 3994;28800 | 4744;14400 | 4744;28800 | 8952;14400 | 8952;28800 |
| | EN | 2540; 2555 | 2540; 2552 | 2551;14400 | 2551;28800 | 2540;14400 | 2540;28800 | 2551;14400 | 2551;28800 | 2540;14400 | 2540;28800 |
| | ES | 2371; 2508 | 2371; 2508 | 2382;14400 | 2382;28800 | 2371;14400 | 2371;28800 | 2382;14400 | 2382;28800 | 2371;14400 | 2371;28800 |
| Pn55k7 | E | 13767;14400 | 13777;28800 | 13779;14400 | 13845;28800 | 13846;14400 | 13777;28800 | 13772;14400 | 13767;28800 | 13777;14400 | 14075;28800 |
| | EN | 14393;14400 | 14393;28800 | 14463;14400 | 14463;28800 | 14393;14400 | 14393;28800 | 14463;14400 | 14463;28800 | 14393;14400 | 14393;28800 |
| | ES | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 | 0;14400 | 0;28800 |
| Pn55k8 | E | 12556;14400 | 12549;22630 | 12548;14400 | 12611;28800 | 12606;14400 | 12549;28800 | 12559;14400 | 12611;28800 | 12549;14400 | 12549;28800 |
| | EN | 12446;14400 | 12446;23441 | 12507;14400 | 12507;28800 | 12446;14400 | 12446;28800 | 12507;14400 | 12507;28800 | 12446;14400 | 12446;28800 |

Continued on next page

| T.c. | Ag. | 1.05 | | 1.10 | | 1.15 | | 1.20 | | ∞ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h | 4h | 8h |
| | ES | 12515;14400 | 22461;24336 | 12558;14400 | 22573;28800 | 12542;14400 | 22461;28800 | 12506;14400 | 22573;28800 | 12494;14400 | 22461;28800 |

Table B.4.: Runtime (in seconds) for the three algorithms and 54 test instances, 5 different bounds for $f_1$ and different maximal runtime (4h,8h ). Runtime till last found solution and till it is proven that the Pareto-set is complete are shown, 0 indicates that the corresponding algorithm could not find the last point found by any other algorithm (E = EPS, EN = EPSN, ES = EPSS)

### B.1.3. Average Runtime Difference

| Bound | 4h | | | 8h | | |
|---|---|---|---|---|---|---|
| | EPSN to EPS | EPSS to EPS | EPSN to EPSS | EPSN to EPS | EPSS to EPS | EPSN to EPSS |
| 1.05 | 3242.51;1420.37 | 831.40;614.07 | 293.92; 66.65 | 3134.04;1412.01 | 813.45;605.79 | 266.80; 67.52 |
| 1.05 | 51.47; 7.64 | 26.34; 15.47 | 20.82; -5.00 | 44.81; 7.79 | 20.92; 15.59 | 20.54; -4.99 |
| 1.05 | 15.82; 4.07 | 11.32; 9.43 | 5.93; -3.47 | 15.64; 4.24 | 10.86; 10.06 | 5.49; -3.91 |
| 1.05 | -2.79; 0.00 | 1.47; 0.00 | -5.75; 0.00 | -15.30; -2.89 | 3.97; 0.00 | -13.43; -2.89 |
| 1.10 | 2713.90; 454.53 | 538.04;244.24 | 276.98; 15.76 | 2632.24; 460.20 | 527.88;244.49 | 234.55; 16.53 |
| 1.10 | 4.95; 12.61 | 21.52; 11.63 | -6.11; -2.96 | 19.01; 12.23 | 21.95; 11.49 | 5.03; -3.01 |
| 1.10 | -3.42; -3.98 | -7.42; 0.49 | 5.69; -3.37 | -15.50; -2.75 | -13.22; -8.91 | -3.47; 7.28 |
| 1.10 | -26.07; 0.00 | -15.79; 0.00 | -13.62; 0.00 | -22.18; 0.00 | -2.72; 0.00 | -13.29; 0.00 |
| 1.15 | 2345.90; 22.44 | 479.22; 20.80 | 232.82; -4.02 | 2280.31; 22.68 | 470.72; 20.31 | 201.37; -3.57 |
| 1.15 | 7.63; -6.75 | 1.58; 3.60 | 8.56; -8.35 | 10.26; -5.64 | 5.13; 3.59 | 6.86; -7.13 |
| 1.15 | -20.78; 0.00 | -17.67; 0.00 | -1.51; 0.00 | -30.36; -3.07 | -25.01; 0.00 | -3.92; -3.07 |
| 1.15 | -29.69; 0.00 | -21.11; 0.00 | -9.60; 0.00 | -19.55; 0.00 | 0.83; 0.00 | -13.42; 0.00 |
| 1.20 | 2340.80; 11.40 | 475.71; -0.57 | 232.82; 7.18 | 2275.21; 10.79 | 467.21; -0.52 | 196.93; 6.97 |
| 1.20 | -3.86; -3.72 | -9.45; -0.50 | 4.75; -3.19 | -3.90; 14.17 | -10.29; 15.60 | 6.45; -0.13 |
| 1.20 | -28.86; 0.00 | -28.16; 0.00 | 3.34; 0.00 | -37.02; -4.26 | -33.47; 0.00 | -2.29; -4.26 |
| 1.20 | -35.16; 0.00 | -26.68; 0.00 | -9.56; 0.00 | -20.43; 0.00 | -2.30; 0.00 | -10.61; 0.00 |
| $\infty$ | 2335.00; 12.28 | 476.92; 6.04 | 224.43; 3.18 | 2274.90; -14.01 | 469.33; -10.08 | 191.79; 0.45 |
| $\infty$ | -28.41; 0.00 | -31.29; 0.00 | 20.12; 0.00 | -33.05; 0.00 | -31.69; 0.00 | 10.23; 0.00 |
| $\infty$ | -55.80; 0.00 | -52.63; 0.00 | -6.56; 0.00 | -49.92; 0.00 | -45.72; -4.16 | -2.49; 8.32 |
| $\infty$ | -39.14; -0.88 | -32.68; -7.40 | -6.96; 58.19 | -22.97; 0.00 | -8.79; 0.00 | -7.04; 0.00 |

Table B.5.: Average runtime difference (in %), shown for 4 groups of instances within each bound for $f_1$. Groups are created by sorting the instances within each bound in increasing order of the runtime needed by the EPS algorithm, and than partitioning them into groups of equal sizes. Average runtime differences till last found solution and till it is proven that the Pareto-set is complete are shown

# C. Acknowledgment

I want to thank

- everyone who made it possible for me to complete my education thus enabling me to write this thesis; especially my family for their support and encouragement; my mother Brigitte Schnellrieder, my father Manfred Reiter, my stepfather Mag. Franz Schnellrieder for all the skills they taught and handed on to me; for the freedom to make my own decisions and the moral support of them; my girlfriend Mag. Susanne Mayr for her understanding and good discussions.

- Ao. Univ.-Prof. Mag. Dr. Walter Gutjahr for supervising this thesis and sparking my interest in operations research/decision support.

- my friends and colleagues at university for their contributions to my studies.

The results of this thesis were obtained by projects funded by the Austrian Science Fund (FWF) grant L264-N13 and P20342.

# Bibliography


1. N.R. Achuthan, L. Caccetta, and S.P. Hill. A new subtour elimination constraint for the vehicle routing problem. *European Journal of Operational Research*, 91:573–586, 1996.

2. E. Alba and J. Francisco Chicano. Software project management with GAs. *Information Sciences*, 177(11):2380–2401, 2007.

3. F. Almeida, M.J. Blesa Aguilera, C. Blum, M. J. Moreno-Vega, M. Pérez Pérez, A. Roli, and Sampels M., editors. *Hybrid Metaheuristics, Third International Workshop, HM 2006, Gran Canaria, Spain, October 13-15, 2006, Proceedings*, volume 4030 of *Lecture Notes in Computer Science*, 2006. Springer. ISBN 3-540-46384-4.

4. A. Arbel. Approximate articulation of preference and priority derivation. *European Journal of Operational Research*, 43(3):317–326, 1989.

5. P. Armand and C. Malivert. Determination of the efficient set in multiobjective linear programming. *Journal of Optimization Theory and Applications*, 70(3):467–489, 1991.

6. D. Armbruster, E.S. Gel, and J. Murakami. Bucket brigades with worker learning. *European Journal of Operational Research*, 176(1):264–274, 2007.

7. D. Armbruster, E.S. Gel, and J. Murakami. Bucket brigades with worker learning. *European Journal of Operational Research*, 176(1):264–274, 2007.

8. N. Ascheuer, M. Fischetti, and M. Grötschl. A Polyhedral Study of the Asymmetric Traveling Salesman Problem with Time Windows. *Networks*, 36(2):69–79, March 2000.

9. N. Ascheuer, M. Fischetti, and M. Grötschl. Solving the Asymmetric Travelling Salesman Problem with time windows by branch-and-cut. *Mathematical Programming*, 90:475–506, 2001.

10. D.T. Asher. A linear programming model for the allocation of R and D efforts. *IRE Transactions on Engineering Management*, 9(4):154–157, 1962.

11. P. Augerat, J.M. Belenguer, E. Benavant, A. Corberán, D. Naddef, and G. Rinaldi. Computational results with a branch and cut code for the capacitated vehicle routing problem. Technical Report RR040-M, Université Joseph Fourier, Grenoble, France, 1995.

12. P. Augerat, J.M. Belenguer, E. Benavant, A. Corberán, and D. Naddef. Seperating capacity inequalities in the CVRP using tabu search. *European Journal of Operational Research*, 106:546–557, 1999.

13. M.A. Badri, D. Davis, and D. Davis. A comprehensive 0-1 goal programming model for project selection. *International Journal of Project Management*, 19(4):243–252, 2001.

14. R. Baldacci, Hadjiconstantinou, and A. Mingozzi. An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation. *Operations Research*, 52(5):723–738, 2004.

15. AG Beged-Dov. Optimal assignment of R&D projects in a large company using an integer programming model. *IEEE Transactions on Engineering Management. EM-12*, pages 138–142, 1965.

16. T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

17. V. Belton and T.J. Stewart. *Multiple criteria decision analysis: an integrated approach.* Springer, 2001.

18. J.-F. Bérubé, M. Gendreau, and J.-Y. Potvin. An exact $\epsilon$-constraint method for bi-objective combinatorial problems: application to the traveling salesman problem with profits. *European Journal of Operational Research*, 194:39–50, 2009.

19. J.R. Birge and F. Louveaux. *Introduction to stochastic programming.* Springer Verlag, 1997.

20. U. Blassum and W. Hochstättler. Application of the branch and cut method to the vehicle routing problem. Technical Report ZPR2000-36. Technical report, ZPR, Universität zu Köln, 2000. Available at http://www.zaik.uni-koeln.de/paper.

21. F.P. Brooks Jr. *The mythical man-month (anniversary ed.)*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1995.

22. J. Butler, D.J. Morrice, and P.W. Mullarkey. A multiple attribute utility theory approach to ranking and selection. *Management Science*, 47(6):800–816, 2001.

23. R. Caballero, T. Gomez, M. Luque, F. Miguel, and F. Ruiz. Hierarchical generation of pareto optimal solutions in large-scale multiobjective systems. *Computers and operations research*, 29(11):1537–1558, 2002.

24. L. Chalmet et al. An algorithm for the bi-criterion integer programming problem. *European Journal of Operational Research*, 25(2):292–300, 1986.

25. A.N.K. Chen and T.M. Edgington. Assessing value in organizational knowledge creation: considerations for knowledge workers. *Management Information Systems Quarterly*, 29(2):6, 2005.

26. F. Chu, N. Labadi, and C. Prins. A scatter search for the periodic capacitated arc routing problem. *European Journal of Operational Research*, 169(2):586–605, 2006.

27. T.H. Cormen. *Introduction to algorithms*. The MIT press, 2001.

28. G. Dantzig and J. Ramsey. The truck dispatching problem. *Management Science*, 6: 80–91, 1959.

29. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley Sons, LTD, 2001.

30. K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In *First International Conference on Evolutionary Multi-Criterion Optimization (EMO-2001)*, 2001.

31. K. Deb and S. Jain. Running performance metrics for evolutionary multi-objective optimization. *KanGAL Report*, 2002004, 2002.

32. K. Deb, A. Pratap, and T. Meyarivan. Constrained Test Problems for Multi-Objective Evolutionary Optimization. In *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 284–298. Springer Verlag, 2000.

33. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2): 182 – 197, 2002.

34. K. Doerner, W.J. Gutjahr, R.F. Hartl, C. Strauss, and C. Stummer. Ant Colony Optimization in Multiobjective Portfolio Optimization. In *4th Methaheuristic International Conference*, pages 243–248, 2001.

35. K. Doerner, W.J. Gutjahr, R.F. Hartl, C. Strauss, and C. Stummer. Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research*, 131(1):79–99, 2004.

36. M. Dorigo and T. Stützle. *Ant colony optimization.* the MIT Press, 2004.

37. H.L. Dreyfus, T. Anthanasiou, and S.E. Dreyfus. *Mind over machine: The power of human intuition and expertise in the era of the computer.* Simon & Schuster, 2000.

38. I. Dumitrescu and T. Stützle. Combinations of local search and exact algorithms. *Applications of Evolutionary Computing*, pages 57–68, 2003.

39. HA Eiselt and V. Marianov. Employee positioning and workload allocation. *Computers and Operations Research*, 35(2):513–524, 2008.

40. J. Erpenbeck and V. Heyse. Kompetenzbiographie–Kompetenzmilieu–Kompetenztransfer. *Zum biographischen Kompetenzerwerb der mittleren Führungsebene, nachgeordneten Mitarbeitern und Betriebsräten. QUEM-report. Schriften zur beruflichen Weiterbildung*, 62, 1999.

41. M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003.

42. G.S. Fishman. *Monte Carlo: concepts, algorithms, and applications.* Springer, 1996.

43. C. M. Fonseca, V. Grunert da Fonseca, and L. Paquete. Exploring the performance of stochastic multiobjective optimisers with the second-order attainment function. In C. C. Coello, A. H. Aguirre, and E. Zitzler, editors, *Evolutionary Multi-criterion Optimization (EMO 2005)*, volume 3410 of *Lecture Notes in Computer Science*, pages 250–264. Springer Verlag, 2005.

44. P. Francis, K. Smilowitz, and M. Tzur. The period vehicle routing problem with service choice. *Transportation Science*, 40(4):439–454, 2006.

45. M.C. Fu. Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, 14(3):192–215, 2002.

46. R. Fukasawa, H. Longo, J. Lysegaard, M. Poggi de Aragao, M. Reis, E. Uchoa, and R. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106(3):491–511, 2006.

47. S.A. Gabriel, S. Kumar, J. Ordóñez, and A. Nasserian. A multiobjective optimization model for project selection with probabilistic considerations. *Socio-Economic Planning Sciences*, 40(4):297–313, 2006.

48. F. Glover and G.A. Kochenberger. *Handbook of metaheuristics*. Springer, 2003.

49. R.R. Greenberg and T.R. Nunamaker. Integrating the analytic hierarchy process (AHP) into the multiobjective budgeting models of public sector organizations. *Socio-Economic Planning Sciences*, 28(3):197–206, 1994.

50. V. Grunert da Fonseca, C.M. Fonseca, and A.O. Hall. Inferential performance assessment of stochastic optimisers and the attainment function. *Lecture notes in computer science*, pages 213–225, 2001.

51. W. Gutjahr. Two metaheuristics for multiobjective stochastic combinatorial optimization. *Stochastic Algorithms: Foundations and Applications*, pages 116–125, 2005.

52. W.J. Gutjahr. Optimal dynamic portfolio selection for projects under a competence development model. *OR Spectrum*, pages 1–34.

53. W.J. Gutjahr. On the finite-time dynamics of ant colony optimization. *Methodology and Computing in Applied Probability*, 8(1):105–133, 2006.

54. W.J. Gutjahr. A provably convergent heuristic for stochastic bicriteria integer programming. *Journal of Heuristics*, 15(3):227–258, 2009.

55. W.J. Gutjahr and P. Reiter. Bi-objective project portfolio selection and staff assignment under uncertainty. *Optimization*, 59(3):417–445, 2010.

56. W.J. Gutjahr, S. Katzensteiner, and P. Reiter. A VNS algorithm for noisy problems and its application to project portfolio analysis. *Lecture notes in computer science*, 4665:93, 2007.

57. W.J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Competence-driven project portfolio selection, scheduling and staff assignment. *Central European Journal of Operations Research*, 16(3):281–306, 2008.

58. W.J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Multi-objective decision analysis for competence-oriented project portfolio selection. *European Journal of Operational Research*, 2010.

59. Y. Haimes, L. Lasdon, and D. Wismer. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, 1:296–297, 1971.

60. C. Heimerl and R. Kolisch. Work assignment to and qualification of multi-skilled human resources under knowledge deprecation and company skill level targets. Technical report, TUM Business School; Technical University Munich, 2008.

61. C. Heimerl and R. Kolisch. Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectrum*, pages 1–26, 2008.

62. M. Held and R.M. Karp. The Traveling Salesman Problem and Minimum Spanning Trees. *Operations Research*, 18:1135–1162, 1970.

63. A.D. Henriksen and A.J. Traynor. A Practical R & D Project-Selection Scoring Tool. *IEEE Transactions on Engineering Management*, 46(2):158–170, 1999.

64. S.W. Hess. A dynamic programming approach to R and D budgeting and project selection. *IRE Transactions on Engineering Management*, 9:170 – 179, 1962.

65. H.H. Hoos and T. Stützle. *Stochastic local search: Foundations and applications.* Morgan Kaufmann, 2005.

66. HR-XML Consortium. Competencies (Measurable Characteristics) Recommendation 2006-02-28. URL `http://ns.hr-xml.org`.

67. C.-L. Hwang and A. S. M. Masud. *Multiple Objective Decision Making - Methods and Applications: A State-of-the-art Survey.* Springer-Verlag, Berlin, 1979.

68. S. Irnich, B. Funke, and T. Gr
"unert. Sequential search and its application to vehicle-routing problems. *Computers and Operations Research*, 33(8):2405–2429, 2006.

69. A. Jaszkiewicz. Evaluation of multiple objective metaheuristics. *Metaheuristics for Multiobjective Optimisation*, 535:65–89, 2004.

70. N. Jozefowiez, F.Semet, and E.G. Talbi. Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem. In J. Guervos,

editor, *Parallel Problem Solving from Nature - PPSN VII*, LNCS. Springer-Verlag, Granada, Spain, 2002.

71. N. Jozefowiez, F. Semet, and E.G. Talbi. Enhancements of NSGA-II and its application to the vehicle routing problem with route balancing. In E. Talbi, editor, *Proceedings of the 7th International Conference Artificial Evolution-EA 2005*, number 3871 in LNCS, pages 131–142. Springer-Verlag, 2006.

72. N. Jozefowiez, F. Semet, and E.G. Talbi. Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293 – 309, 2008.

73. S. Kavadias and C. Loch. *Project selection under uncertainty: dynamically allocating resources to maximize value.* Kluwer Academic Pub, 2004.

74. R. Khorramshahgol and Y. Gousty. Delphic Goal Programming(DGP)- A multi-objective cost/benefit approach to R & D portfolio analysis. *IEEE Transactions on Engineering Management*, 33:172–175, 1986.

75. J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, February 2006.

76. R. Kolisch and S. Hartmann. Experimental investigation of heuristics for resource-constrained project scheduling: An update. *European Journal of Operational Research*, 174(1):23–37, 2006.

77. P. Lacomme, C. Prins, and W. Ramdane-Cherif. Evolutionary algorithms for periodic arc routing problems. *European Journal of Operational Research*, 165(2):535–553, 2005.

78. G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance constrained vehicle routing problem. *Networks*, 14:161–172, 1984.

79. G. Laporte, H. Mercure, and Y. Nobert. An exact algorithm for the asymmetrical capacitated vehicle routing problem. *Networks*, 16, 1986.

80. J.B. Lassiter, M.M. Wiecek, and K.R. Andrighetti. Lagrangian coordination and analytical target cascading: solving ATC-decomposed problems with Lagrangian duality. *Optimization and Engineering*, 6(3):361–381, 2005.

81. M. Laumanns, L. Thiele, and E. Zitzler. An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. *European Journal of Operational Research*, 169(3), March 2006.

82. J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11:221–227, 1981.

83. J. Liesiö, P. Mild, and A. Salo. Preference programming for robust portfolio modeling and project selection. *European Journal of Operational Research*, 181(3):1488–1505, 2007.

84. C.H. Loch and S. Kavadias. Dynamic portfolio selection of NPD programs using marginal returns. *Management Science*, 48(10):1227–1241, 2002.

85. J. Lysgaard. CVRPSEP: A package of seperations routines for the capacitated vehicle routing problem. *http://www.hha.dk/ lys*.

86. J. Lysgaard, A. N. Letchford, and R. W. Eglese. A New Branch-and-Cut Algorithm for the Capacitated Vehicle Routing Problem. *Mathematical Programming*, 100:2004, 2003.

87. R.S. Mansfield. Building competency models: Approaches for HR professionals. *Human Resource Management*, 35(1):7–18, 1996.

88. A.L. Medaglia, S.B. Graves, and J.L. Ringuest. A multiobjective evolutionary approach for linearly constrained project selection under uncertainty. *European Journal of Operational Research*, 179(3):869–894, 2007.

89. Z. Michalewicz and J. Arabas. Genetic algorithms for the 0/1 knapsack problem. *Methodologies for Intelligent Systems*, pages 134–143, 1994.

90. K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

91. R.H. Möhring and F. Stork. Linear preselective policies for stochastic project scheduling. *Mathematical Methods of Operations Research*, 52(3):501–515, 2000.

92. G.L. Nemhauser and L.A. Wolsey. *Integer and combinatorial optimization*. Wiley New York, 1999.

93. O. Ngwenyama, A. Guergachi, and T. McLaren. Using the learning curve to maximize IT productivity: A decision analysis model for timing software upgrades. *International Journal of Production Economics*, 105(2):524–535, 2007.

94. C.H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity.* Dover Pubns, 1998.

95. J.M. Pasia, K. F. Doerner, R. F. Hartl, and M. Reimann. A population-based local search for solving a bi-objective vehicle routing problem. volume 4446 of *LNCS*, pages 166–175. EvoCOP 2007, 2007.

96. J.M. Pasia, K.F. Doerner, R.F. Hartl, and M. Reimann. Solving a bi-objetive vehicle routing problem by pareto-ant colony optimization. volume 4638 of *LNCS*, pages 187–191. SLS 2007, 2007.

97. P.C. Pendharkar and G.H. Subramanian. An empirical study of ICASE learning curves and probability bounds for software development effort. *European Journal of Operational Research*, 183(3):1086–1096, 2007.

98. G.C. Pflug. *Optimization of stochastic models.* Kluwer, 1996.

99. C. Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31:1985 – 2002, 2004.

100. J. Puchinger and G.R. Raidl. Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. *Artificial Intelligence and Knowledge Engineering Applications: a Bioinspired Approach*, pages 41–53, 2005.

101. G. Raidl. A unified view on hybrid metaheuristics. *Hybrid Metaheuristics*, pages 1–12, 2006.

102. G. R. Raidl. A Unified View on Hybrid Metaheuristics. In *Hybrid Metaheuristics*, pages 1–12, 2006.

103. T.K. Ralphs, L. Kopmann, W.R. Pulleyblank, and L.E. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359, 2003.

104. T.K. Ralphs, M.J. Saltzman, and M.M. Wiecek. An improved algorithm for solving biobjective integer programs. *Annals of Operations Research*, 147(1):43–70, 2006.

105. P. Reiter and W.J. Gutjahr. Exact hybrid algorithms for solving a bi-objective vehicle routing problem. *Central European Journal of Operations Research*, pages 1–25.

106. P. Reiter, W.J. Gutjahr, S. Katzensteiner, and C. Stummer. Multiobjective stochastic project portfolio selection and scheduling using metaheuristics. In *Proceedings of the Metaheuristics International Conference 2007*, 2007.

107. R.Y. Rubinstein and D.P. Kroese. *Simulation and the Monte Carlo method*. Wiley-Interscience, 2008.

108. J.S. Shippmann, R.A. Ash, M. Battista, L. Carr, L.D. Eyde, B. Hesketh, J. Kehoe, K. Pearlman, E.P. Prien, and J.I. Sanchez. The practice of competency modeling. *personnel psychology*, 53(3):703–740, 2000.

109. R.E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. Wiley, 1986.

110. R.E. Steuer. *Multiple Criteria Decision Making*, chapter The ADABASE Multiple Objective Linear Programming Package, pages 1–6. SCI-TECH, Windsor, England, 1995.

111. C. Stummer and K. Heidenberger. Interactive R&D portfolio analysis with project interdependencies and time profiles of multiple objectives. *IEEE Transactions on Engineering Management*, 50(2):175–183, 2003.

112. C. Stummer and M. Sun. New multiobjective metaheuristic solution procedures for capital investment planning. *Journal of Heuristics*, 11(3):183–199, 2005.

113. T. Stützle, H.H. Hoos, et al. MAX-MIN ant system. *Future Generation Computer Systems*, 16(8):889–914, 2000.

114. GA Suer and RR Tummaluri. Multi-period operator assignment considering skills, learning and forgetting in labour-intensive cells. *International Journal of Production Research*, 46(2):469–493, 2008.

115. C.K. Suh, E.H. Suh, and K.C. Baek. Prioritizing telecommunications technologies for long-range R&D planning to the year 2006. *IEEE transactions on engineering management*, 41(3):264–275, 1994.

116. E.G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of heuristics*, 8(5):541–564, 2002.

117. K. Tarvainen and Y.Y. Haimes. Coordination of hierarchical multiobjective systems: theory and methodology. *IEEE Transactions on Systems, Man and Cybernetics*, 12 (6):751–764, 1982.

118. P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.

119. F. Tricoire, K.F. Doerner, R.F. Hartl, and M. Iori. Heuristic and Exact Algorithms for the Multi-Pile Vehicle Routing Problem. Submitted to OR Spectrum.

120. TSPLIB. URL `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`.

121. C. L. Valenzuela and A. J. Jones. Estimating the Held-Karp lower bound for the geometric TSP. *European Journal of Operational Research*, 102(1):157–175, 1997.

122. T. Volgenant and R. Jonker. A branch and bound algorithm for the symmetric traveling salesman problem based on 1-tree relaxation. *European Journal of Operational Research*, 9:83–89, 1982.

123. T. Wagner, N. Beume, and B. Naujoks. Pareto-, aggregation-, and indicator-based methods in many-objective optimization. In *Evolutionary Multi-Criterion Optimization*, pages 742–756. Springer, 2007.

124. M. Weber. Decision making with incomplete information. *European Journal of Operational Research*, 28(1):44–57, 1987.

125. M.C. Wu and S.H. Sun. A project scheduling and staff assignment model considering learning effect. *The International Journal of Advanced Manufacturing Technology*, 28(11):1190–1195, 2006.

126. M. Yoshimura, Y. Fujimi, K. Izui, and S. Nishiwaki. Decision-making support system for human resource allocation in product development projects. *International journal of production research*, 44(5):831–848, 2006.

127. E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithmsA comparative case study. In *Parallel Problem Solving from NaturePPSN V*, pages 292–301. Springer, 1998.

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

129. E. Zitzler, D. Brockhoff, and L. Thiele. The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *Evolutionary Multi-Criterion Optimization (LNCS 4403)*, pages 862–876, Berlin, 2007. Springer.

# Abstract

The research goals of this thesis are the development of algorithms to solve multi-objective and stochastic optimization problems in the field of scheduling and routing problems.

In practice decision problems often include different goals which can hardly be aggregated to a single objective for different reasons. In the field of multi-objective optimization several objective functions are considered. As in single objective optimization a solution has to satisfy all constraints of the problem. In general the goals are conflicting and there will be no solution, that is optimal for all objectives. Algorithms for multi-objective optimization problems provide the decision maker a set of efficient solutions, among which she or he can choose the most suitable alternative. In multi-objective optimization efficiency of a solution is expressed as Pareto-optimality. Pareto-optimality of a solution is defined as the property that no other solution exists that is better than the proposed one in at least one objective and at least equally good in all criteria.

The first application that is considered in this thesis, the *Multi-objective Project Selection, Scheduling and Staffing with Learning* problem (MPSSSL) arises from the field of management in research-centered organizations. Given a set of project proposals the decision makers have to select the "best" subset of projects (a project portfolio) and set these up properly (schedule them and provide the necessary resources). This problem is hard to solve for different reasons: (i) selecting a subset of projects considering limited resources is a knapsack-type problem that is known to be $\mathcal{NP}$-hard, and (ii) to determine the feasibility of a given portfolio, the projects have to be scheduled and staff must be assigned to them. As in this problem the assignment of workers is influenced by the decision which portfolio should be selected, the decision maker has to consider goals of different nature. Some objectives are related to economic goals (e.g. return of investment), others are related to the competence development of the workers. Competence oriented goals are motivated by the fact that competencies determine the attainment and sustainability of strategic positions in market competition. In general the objectives cannot be combined to a single objective, therefore methods for solving multi-objective optimization problems are used. To solve the problem we use two different hybrid algorithms that combine metaheuristic algorithms, (i) the *Nondominated Sorting Genetic Algorithm* (NSGA-II), and (ii)

*Pareto Ant Colony* (P-ACO) algorithm with a *linear programming solver* as a subordinate.

In practice, uncertainty is another typically encountered aspect. Different parameters of the problem can be uncertain (e.g. benefits of a project, or the time and effort required to perform the single activities required by a project). To determine the "best" portfolio, methods are needed that are able to handle uncertainty in optimization. To solve the stochastic extension (SMPSSSL) of the MPSSSL problem we present an algorithm that combines the aforementioned NSGA-II algorithm with the *Adaptive Pareto Sampling* (APS) algorithm. APS is used to handle the interplay between multi-objective optimization and simulation. The performance of the simulation process is increased by using *importance sampling* (IS).

The second problem, the *Bi-objective Capacitated Vehicle Routing Problem with Route Balancing* (CVRPB) arises from the field of vehicle routing. Given a set of customers, the decision makers have to construct routes for a fixed number of vehicles, each starting and ending at the same depot, such that the demands of all customers can be fulfilled, and the capacity constraints of each vehicle are not violated. The traditional objective of this problem (known as the Capacitated Vehicle Routing Problem (CVRP)) is minimizing the total costs of all routes. A problem that may arise by this approach is that the resulting routes can be very unbalanced (in the sense of drivers workload). To overcome this problem a second objective function that measures the balance of the routes of a solution is introduced. In this work, we use the *Adaptive $\varepsilon$-Constraint Method* in combination with a branch-and-cut algorithm and two genetic algorithms (i) a single-objective GA and (ii) the multi-objective NSGA-II, to solve the considered problem.

Prototypes of different algorithms to solve the problems are developed and their performance is assessed by using state of the art performance measures. The computational experiments show that the developed solution procedures will be well suited to solve the considered optimization problems. The hybrid algorithms combining metaheuristic and exact optimization methods, turned out to be crucial to solve the problem (application to project portfolio selection) or to improve the performance of the solution procedure (application to vehicle routing).

# Abstract in German

Ziel dieser Arbeit ist die Entwicklung von Optimierungsalgorithmen, mit denen Mehrziel- und stochastische Mehrziel-Optimierungsprobleme gelöst werden können.

In der Praxis beinhalten Optimierungsprobleme oft unterschiedliche Ziele, welche optimiert werden sollen. Oft ist es nicht möglich die Ziele zu einem einzelnen Ziel zusammenzufassen. Mehrzieloptimierung beschäftigt sich damit, solche Probleme zu lösen. Wie in der Einzieloptimierung muss eine Lösung alle Nebenbedingungen des Problems erfüllen. Im Allgemeinen sind die Ziele konfligierend, sodass es nicht möglich ist eine einzelne Lösung zu finden welche optimal im Sinne aller Ziele ist. Algorithmen zum Lösen von Mehrziel-Optimierungsproblemen, präsentieren dem Entscheider eine Menge von effizienten Alternativen. Effizienz in der Mehrzieloptimierung ist als Pareto-Optimalität ausgedrückt. Eine Lösung eines Optimierungsproblems ist genau dann Pareto-optimal wenn es keine andere zulässige Lösung gibt, welche in allen Zielen mindestens gleich gut wie die betrachtete Lösung ist und besser in mindestens einem Ziel.

In dieser Arbeit werden Mehrziel-Optimierungsprobleme aus zwei unterschiedlichen Anwendungsgebieten betrachtet. Das erste Problem, das *Multi-objective Project Selection, Scheduling and Staffing with Learning* Problem (MPSSSL), entstammt dem Management in forschungsorientierten Organisationen. Die Entscheider in solchen Organisationen stehen vor der Frage welche Projekte sie aus einer Menge von Projektanträgen auswählen sollen, und wie diese Teilmenge von Projekten (ein Projektportfolio) mit den benötigten Ressourcen ausgestattet werden kann (dies beinhaltet die zeitliche und personelle Planung). Aus unterschiedlichen Gründen ist dieses Problem schwer zu lösen, z.B. (i) die Auswahl von Projekten unter Beachtung der beschränkten Ressourcen ist ein Rucksackproblem (und ist damit $\mathcal{NP}$-schwer) (ii) ob ein Projektportfolio zulässig ist oder nicht hängt davon ab ob, man dafür einen Zeitplan erstellen kann und genügend Mitarbeiter zur Verfügung stehen. Da in diesem Problem die Mitarbeiterzuordnung zu den einzelnen Projekten einbezogen wird, muss der Entscheider Ziele unterschiedlicher Art berücksichtigen. Manche Ziele sind öknomischer Natur, z.B. die Rendite, andere wiederum beziehen sich auf die Kompetenzentwicklung der einzelnen Mitarbeiter. Ziele, die sich auf die Kompetenzentwicklung beziehen, sollen sicherstellen, dass das Unternehmen auch in Zukunft am Markt be-

stehen kann. Im Allgemeinen können diese unterschiedlichen Ziele nicht zu einem einzigen Ziel zusammengefasst werden. Daher werden Methoden zur Lösung von Mehrziel-Optimierungsproblemen benötigt. Um MPSSSL Probleme zu lösen werden in dieser Arbeit zwei unterschiedliche hybride Algorithmen betrachtet. Beide kombinieren nämlich Metaheuristiken (i) den *Nondominated Sorting Genetic* (NSGA-II) Algorithmus, und den (ii) *Pareto Ant Colony* (P-ACO) Algorithmus, mit einem exakten Algorithmus zum Lösen von Linearen Programmen kombinieren.

Unsicherheit ist ein weiterer wichtiger Aspekt der in der Praxis auftaucht. Unterschiedliche Parameter des Problems können unsicher sein (z.B. der aus einem Projekt erzielte Gewinn oder die Zeit bzw. der Aufwand, der benötigt wird, um die einzelnen Vorgänge eines Projekts abzuschließen). Um in diesem Fall das "beste" Projektportfolio zu finden, werden Methoden benötigt, welche stochastische Mehrziel-Optimierungsprobleme lösen können. Zur Lösung der stochastischen Erweiterung (SMPSSSL) des MPSSSL Problems zu lösen, präsentieren wir eine Methode, die den zuvor genannten hybriden NSGA-II Algorithmus mit dem *Adaptive Pareto Sampling* (APS) Algorithmus kombiniert. APS wird verwendet, um das Zusammenspiel von Simulation und Optimierung zu koordinieren. Zur Steigerung der Performance des Simulationsprozesses, verwenden wir *Importance Sampling* (IS).

Das zweite Problem dieser Arbeit, das *Bi-Objective Capacitated Vehicle Routing Problem with Route Balancing* (CVRPB), kommt aus dem Bereich Logistik. Wenn man eine Menge von Kunden zu beliefern hat, steht man als Entscheider vor der Frage, wie man die Routen für eine fixe Anzahl von Fahrzeugen (mit beschränkter Kapazität) bestimmt, sodass alle Kunden beliefert werden können. Die Routen aller Fahrzeuge starten und enden dabei immer bei einem Depot. Die Einziel-Variante dieses Problems ist als *Capacitated Vehicle Routing Problem (CVRP)* bekannt, dessen Ziel es ist die Lösung zu finden, die die Gesamtkosten aller Routen minimiert. Dabei tritt jedoch das Problem auf, dass die Routen der optimalen Lösung sehr unterschiedliche Fahrtzeiten haben können. Unter bestimmten Umständen ist dies jedoch nicht erwünscht. Um dieses Problem zu umgehen, betrachten wir in dieser Arbeit eine Variante des (bezeichnet als CVRPB) CVRP, welche als zweite Zielfunktion die Balanziertheit der einzelnen Routen einbezieht. Zur Lösung von CVRPB Problemen verwenden wir die *Adaptive $\varepsilon$-Constraint Method* in Kombination mit einem Branch-and-Cut Algorithmus und zwei unterschiedlichen Genetischen Algorithmen (GA), (i) einem Einziel-GA und (ii) dem NSGA-II.

In dieser Arbeit werden Optimierungsalgorithmen präsentiert, welche es erlauben, Mehrziel- und stochastische Mehrziel-Optimierungsprobleme zu lösen. Unterschiedliche Algorithmen wurden implementiert und basierend auf aktuellen Performance-Maßen ver-

glichen.
Experimente haben gezeigt, dass die entwickelten Methoden gut geeignet sind, die betrachteten Optimierungsprobleme zu lösen. Die hybriden Algorithmen, welche Metaheuristiken mit exakten Methoden kombinieren, waren entweder ausschlaggebend um das Problem zu lösen (im Fall des Project Portfolio Selection Problems) oder konnten die Performance des Lösungsprozesses signifikant verbessern (im Fall des Vehicle Routing Problems).

# Peter Reiter

*Karl Schönherrstr. 29*
*6300 Wörgl*
✉ *peter.reiter[AT]univie.ac.at*

## Personal data

| | |
|---|---|
| citizenship | Austrian |
| date of birth | February 7, 1982 in Oberndorf (Sbg), Austria |

## Education

| | |
|---|---|
| Mar. 2008 - present | **Doctoral Program of Technical Sciences, Business Informatics**, *University of Vienna*, Vienna, Austria. |
| Oct. 2005 - Feb. 2008 | **Master Program in Business Informatics**, *University of Vienna*, Vienna, Austria. |
| Oct. 2002 - Jul. 2005 | **Bachelor Program in Business Informatics**, *University of Vienna*, Vienna, Austria. |

## PhD Thesis

| | |
|---|---|
| title | *Matheuristic Algorithms for Solving Multi-objective/Stochastic Scheduling and Routing Problems* |
| supervisor | Walter J. Gutjahr |

## Master Thesis

| | |
|---|---|
| title | *The Next Release Problem: An extended model formulation and the comparison of two algorithms for stochastic multi-objective combinatorial optimization problems* |
| supervisor | Walter J. Gutjahr |

## Bachelor thesis

| | |
|---|---|
| title | *Implementierung, Parametrisierung & Vergleich von Heuristiken: Simulated Annealing & Genetischer Algorithmus* |
| supervisor | Walter J. Gutjahr |
| title | *BS 15000: Übersicht, Anwendungen und Modelle* |
| supervisor | Harald Kuehn |

## Experience

### Vocational

| | |
|---|---|
| Jan. 2008 - present | **Research Assistant**, *Project: "Matheuristics: Hybride Algorithmen für Transportprobleme"*, Department of Statistics and Decision Support Systems (ISDS), University of Vienna, Vienna. |

Dec. 2006 - Dec. 2007 **Research Assistant**, *Project: "Kompetenzgesteuerte Projektanalyse"*, Department of Statistics and Decision Support Systems (ISDS), University of Vienna, Vienna.

Oct. 2006 - Feb. 2007 **Tutor**, *Grundlagen Decision Support and Software-Einsatz im Operations Research I*, Department of Statistics and Decision Support Systems (ISDS), University of Vienna, Vienna.

### Teaching

summer term **Operations Research II**, *University of Vienna*, Stochastic optimization, Queuing theory, Inventory models.

**OR-Methoden in Produktion und Logistik I**, *University of Vienna*, Branch & Bound, Dynamic Programming, Facility Location Problems, Vehicle Routing Problems, Network Flow Problems, Scheduling Problems.

winter term **Operations Research I**, *University of Vienna*, Linear Programming and Modeling, Sensitivity Analysis, Duality, Nonlinear Programming and Modeling.

**Computational Techniques**, *University of Vienna*, Combinatorial Optimization, Integer Linear Programming, Branch & Bound, Branch-and-Cut, Greedy Algorithms, Metaheuristics, NP-complete Problems.

## Publications and Reports

W.J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Multi-objective decision analysis for competence-oriented project portfolio selection. *European Journal of Operational Research*, 2010.

W.J. Gutjahr and P. Reiter. Bi-objective project portfolio selection and staff assignment under uncertainty. *Optimization*, 59(3):417–445, 2010.

P. Reiter and W.J. Gutjahr. Exact hybrid algorithms for solving a bi-objective vehicle routing problem. *Central European Journal of Operations Research*, pages 1–25, 2010.

W.J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Competence-Driven Project Portfolio Selection, Scheduling and Staff Assignment. *Central European J. of Operations Research*, 16:281–306, 2008.

W.J. Gutjahr, S. Katzensteiner, and P. Reiter. A VNS algorithm for noisy problems and its application to project portfolio analysis. In J. Hromkovic et al., editor, *Proc. SAGA 2007 (Stochastic Algorithms: Foundations and Applications)*, volume 4665 of *Springer Lecture Notes in Computer Science*, pages 93 – 104, 2007.

## Talks & Presentations

2010 Reiter P. and Gutjahr W.J. (2010): "Hybrid Algorithms for Solving a Bi-objective Stochastic Periodic Vehicle Routing Problem", Matheuristics 2010: third international workshop on model-based metaheuristics, Vienna, Austria

2009 Reiter P. and Gutjahr W.J. (2009): "An exact and a matheuristic algorithm for the Vehicle Routing Problem with Route Balancing", Annual Conference Italien Operational Research Society, Siena, Italy

2008 Reiter P. (2008): "The Next Release Problem",ÖGOR 30th Anniversary Meeting and General Meeting 2008 , Vienna, Austria

2007 Reiter P., Gutjahr W.J., Katzensteiner S., and Stummer C. (2007): "Multi-objective Stochastic Project Portfolio Selection and Scheduling Using Metaheuristics", Metaheuristic International Conference '07, Montreal, Canada

## Languages

german **mother tongue**

english **fluent in written and spoken**

## Computer skills

| | | | |
|---|---|---|---|
| programming | C++, Java, PHP, SQL, HTML | math packages | Scilab, R, SPSS |
| optimization | Cplex, XpressMP, SCIP | datamining | SPSS Clementine |