MULTI-OBJECTIVE ANALYSIS OF APPROACHES TO DYNAMIC ROUTING OF A VEHICLE

Complete Research

Grimme, Christian, University of Münster, Germany, Christian.Grimme@wi.uni-muenster.de

Meisel, Stephan, University of Münster, Germany, Stephan.Meisel@uni-muenster.de

Trautmann, Heike, University of Münster, Germany, Trautmann@uni-muenster.de

Rudolph, Guenter, TU Dortmund University, Germany, Guenter.Rudolph@tu-dortmund.de

Wölck, Martin, University of Münster, Germany, Martin.Woelck@uni-muenster.de

Abstract

We consider a routing problem for a single vehicle serving customer locations in the course of time. A subset of these customers must necessarily be served, while the complement of this subset contains dynamic customers which request for service over time, and which do not necessarily need to be served. The decision maker's conflicting goals are serving as many customers as possible as well as minimizing total travel distance. We solve this bi-objective problem with an evolutionary multi-objective algorithm in order to provide an a-posteriori evaluation tool for enabling decision makers to assess the single-objective solution strategies that they actually use in real-time. We present the modifications to be applied to the evolutionary multi-objective algorithm NSGA2 in order to solve the routing problem, we describe a number of real-time single-objective solution strategies, and we finally use the gained efficient trade-off solutions of NSGA2 to exemplarily evaluate the real-time strategies. Our results show that the evolutionary multi-objective approach is well-suited to generate benchmarks for assessing dynamic heuristic strategies. Our findings point into future directions for designing dynamic multi-objective approaches for the vehicle routing problem with time windows.

Keywords: multi-objective optimization, evolutionary algorithms, vehicle routing.

1 Introduction

We propose a new approach to solving a multi-objective vehicle routing problem with time windows. This approach serves for analyzing the performance of dynamic routing strategies, i.e., of routing strategies that are applied to the online version of the considered vehicle routing problem. Our main goal is to provide real-time routing decision makers with a tool for assessing their decisions in retrospect. In an online setting with multiple objectives such an a-posteriori analysis may be a valuable source of information for improving the decision maker's understanding of his dynamic routing strategies.

In particular the considered routing problem comprises one single vehicle and a number of customer locations. Part of the customer locations are known in advance and must necessarily be visited before the vehicle reaches its end depot. The remaining customers become known only after having issued a request at a certain point in time. Serving these customers is not obligatory, and the decision maker has to make a trade-off between two conflicting objectives. On the one hand the pursued goal is to serve as many customers as possible, whereas, on the other hand, the total distance travelled must be minimized.

For the a-posteriori version of the routing problem, where all points in time of customer requests are known, we develop a new multi-objective evolutionary algorithm. The Pareto-set of solutions returned by this algorithm is then used as a benchmark for solutions generated from application of dynamic routing strategies to the corresponding online problem.

The remainder of this paper is organized as follows. Section 2 provides an overview of related approaches to multi-objective routing of a vehicle with optional customer visits. Section 3 contains a detailed description of the routing problem considered. In Section 4 we describe our multi-objective evolutionary algorithm for solving the a-posteriori problem. The following Section 5 presents three alternative dynamic strategies for solving the online version of the routing problem, where decisions are made over time while new customer requests occur. In Section 6 test problem instances and algorithmic performance measures are defined. Section 7 provides computational results in terms of both comparisons of dynamic routing strategies and comparisons of dynamic routing strategies with the evolutionary algorithm. Section 8 concludes the paper.

2 Related Work

The present work is related to a number of neighboring lines of research. Traveling Salesman Problems where part of the customer locations do not necessarily have to be included into the vehicle's route are often referred to as "Orienteering Problems" (e.g., Golden et al., 1987), as "Selective Traveling Salesman Problems" (e.g., Gendreau et al., 1998; Laporte and Martello, 1990), or as "Traveling Salesman Problems with Profits" (e.g., Feillet et al., 2005). For the sake of convenience we are going to only use the term "Orienteering Problem" in the remainder of this work.

The case of customer locations not only being optional, but also having a point in time from which on they may be visited by the vehicle falls into the problem class of orienteering problems with time windows. However, the problem described in Section 3 differs from traditional orienteering problems with time windows (e.g., Kantor and Rosenwein, 1992) in two important respects. On the one hand, the lower bound of a customer's time window does not represent the earliest point in time at which the vehicle may arrive at this customer's location. Instead, the lower bound of a customer's time window represents the earliest point in time at which the vehicle may leave the preceding customer location.

On the other hand, our problem formulation differs from the vast majority of publications on orienteering problems in terms of the pursued objective. Although it has been recognized (e.g., Feillet et al., 2005) that orienteering problems should be viewed as bicriteria optimization problems, most existing works only consider problem formulations with the single objective of maximizing the number of customers visited within a given amount of time. In contrast, we consider a proper bicriteria problem formulation that additionally considers the objective of minimizing the total distance traveled by the vehicle.

Keller and Goodchild (1988) were the first to formulate a proper bicriteria model of an orienteering problem. Time windows are not considered, and heuristic solutions are determined by applying a multi-step algorithm that is found to perform well for instances of up to 25 customer locations. The early work of Dell'Amico et al. (1995) comprises two variants of orienteering problems without time windows. Each of the variants aims at minimization of the sum of total distance traveled and penalties for unvisited customers. The resulting single objective problems are formulated as integer programs for which lower bounds are derived.

Feillet et al. (2005) elaborate on the fact that orienteering problems are bicriteria problems by definition. They then conclude that so far virtually all authors avoid dealing with true bicriteria formulations, but rely on either penalty terms or additional constraints for implicitly taking into account a second objective with a single objective model. Against this background, they provide an extensive survey of approaches to a variety of orienteering problems.

Both Berube et al. (2009) and Filippi and Stevanato (2013) rely on solving a series of single-objective optimization problems for approximating the Pareto-frontier of the bi-objective orienteering problem without time windows. Berube et al. (2009) propose an ε-constraint method that repeatedly transforms one of the two objectives into a constraint of a single-objective problem to be solved. This approach is shown to be able to generate the exact Pareto-frontier for the problem, provided that the single-objective problems are solved by branch-and-cut methods with specific improvement heuristics. The approximation schemes proposed by Filippi and Stevanato (2013) are proven to provide a Pareto-ε-approximation of the efficient set of solutions. The authors show their approach to be more efficient than exact approaches to determining the efficient set.

In contrast to the two preceding works Jozefowiez et al. (2008) propose an approach for explicitly solving the bicriteria formulation of the orienteering problem without time windows. The approach determines a high quality approximation of the efficient Pareto-frontier by two main steps. First an NSGA2 based multi-objective evolutionary algorithm is applied for generating a set of initial solutions. These solutions serve as starting points for an ejection chain process with two sets of neighborhood moves. A computational comparison with an iterated ε-constraint implementation of a state-of-the-art meta-heuristic for the single-objective orienteering problem shows that the method has advantages as the problem size increases.

To our knowledge, the approach of Jozefowiez et al. (2008) so far is the only one that explicitly solves the bicriteria version of an orienteering problem. Therefore it may be considered as most related to the approach we present in the following sections. Our approach, however, is tailored to orienteering problems with time windows as described in the following Section 3.

3 Problem Formulation

The vehicle routing problem we consider comprises one vehicle and a set $C = \{1,2,\ldots,N\}$ of geographic locations i, where $i = 1$ represents the start depot and $i = N$ represents the end depot of the vehicle. All remaining locations represent customers that the vehicle may visit. Travel distances d_{ij} between any pair (i,j) of locations are assumed to be both known and deterministic. We assume that one unit of travel distance is corresponding to one unit of time. The vehicle is located at $i = 1$ only at the beginning of its tour, and it must reach the end depot for the first time only at the end of its tour.

The set $C \setminus \{1,N\}$ of customers consists of two complementary parts C^a and C^d. C^a represents customers that have requested for service before the vehicle has left the start depot, and that must be visited once before the vehicle reaches N. In contrast C^d represents a set of dynamic customers, where each customer i issues one service request at a particular point in time t_i. Due to the attempt to reduce travel time, it may happen that a service request of a dynamic customer gets rejected. Note that in practice "rejection" of a request typically implies this request either being postponed to the next day or being shifted to another vehicle's route. However, once a dynamic service request is not rejected but confirmed, the issuing customer must be visited before the vehicle reaches the end depot.

Decision makers face two opposite criteria when dealing with the type of routing problem described. Clearly, all customer requests should be confirmed and visited in order to maximize customer satisfaction. On the other hand, however, satisfying all requests typically implies high costs in terms of a long distance route. As a driver's working shift typically features a certain amount of flexibility, decision makers are interested in making an appropriate trade-off between the two goals of maximizing customer satisfaction and minimizing total routing distance.
In practice both decisions about confirmation (rejection) of customer requests and routing decisions are made in real-time. Customer requests are not known in advance but are revealed gradually over time while the vehicle is operated. In such a situation, decision makers typically rely on decision rules that they apply each time a new customer request occurs (see , e.g., Meisel, 2011). Such a real-time approach to vehicle routing necessarily suffers from the fact that decisions are made without all information about customer requests being known. As a consequence, optimal decisions will rarely be made, and the problem of a-priori selection of the best decision rule is hard.
However, decision makers may gradually gain intuition on the performance of decision rules by assessing the applied rule in retrospect. The remainder of the present section describes the bicriteria problem to be solved in order to determine an ideal a-posteriori solution. In particular, we consider the problem as described by the the following Equations 1a-1h:

$$\min\Big((|C|-\sum_{i\in C}x_i^c),(\sum_{(i,j)\in E}d_{ij}x_{ij}^r)\Big) \tag{1a}$$

$$\begin{aligned}
s.t. \quad & x_i^c &=\;& 1 & \forall i\in C\setminus\{C^d\}, \quad & \text{(1b)}\\
& \sum_{(i,j)\in\phi^+(i)} x_{ij}^r &=\;& x_i^c & \forall i\in C\setminus\{N\}, \quad & \text{(1c)}\\
& \sum_{(k,i)\in\phi^-(i)} x_{ki}^r &=\;& x_i^c & \forall i\in C\setminus\{1\}, \quad & \text{(1d)}\\
& \sum_{(k,i)\in\phi^-(i)} y_{ki}+\sum_{(k,i)\in\phi^-(i)} d_{ki}x_{ki}^r &\le\;& \sum_{(i,j)\in\phi^+(i)} y_{ij} & \forall i\in C\setminus\{1\}, \quad & \text{(1e)}\\
& \sum_{(i,j)\in\phi^+(i)} t_i x_{ij}^r+\sum_{(k,i)\in\phi^-(i)} d_{ki}x_{ki}^r &\le\;& \sum_{(i,j)\in\phi^+(i)} y_{ij} & \forall i\in C, \quad & \text{(1f)}\\
& y_{ij}\in\mathbb{N}_0 & & & \forall (i,j)\in E, \quad & \text{(1g)}\\
& x_{ij}^r, x_i^c\in\{0,1\} & & & \forall (i,j)\in E, \forall i\in C \quad & \text{(1h)}
\end{aligned}$$

Three types of decision variables are involved. For each location $i \in C$, decision x_i^c indicates whether or not i must be included in the vehicle's route, i.e., for each customer i decision x_i^c indicates whether or not the request of i has been confirmed. Moreover, decisions x_{ij}^r determine whether or not the road link connecting locations i and j is part of the vehicles ideal route. Note that we assume that all locations of C are connected with each other, and that distances between any pair i, j of locations are symmetric. The set of all road links connecting any pair (i, j) is referred to as E. The Set $\phi^+(i)$ (the set $\phi^-(i)$) contains all road links that are outgoing from (ingoing into) customer i. Each of the variables y_{ij} is equal to zero if link $(i, j) \in E$ is not part of the solution, i.e., $y_{ij} = 0$ if $x_{ij}^r = 0$. In case of $x_{ij}^r = 1$, variable y_{ij} represents the point in time at which the vehicle arrives at location i.
As stated by Equation 1a we aim at minimization of both the number of unserved customers and the total distance traveled by the vehicle. Equation 1b guarantees that both depots as well as all advance customers $i \in I^a$ will be visited before returning to N. Equations 1c and 1d ensure that the vehicle leaves the start depot, reaches the end depot and visits each customer location with a confirmed request exactly once.
Equation 1e makes sure that the vehicle never arrives at location i before it's predecessor k (according to the route) has been visited and the distance d_{ki} has been traveled. As a consequence we have $y_{ki} \le y_{ij}$ with j representing the successor of i. Additionally, Equation 1f ensures that the vehicle is never allowed to

leave the preceding location k for moving on to i before i has actually issued a service request. Note that the claim of Equation 1f reflects the conditions present in a real-time setting.
From the point of view for the vehicle routing literature, the proposed model represents a bicriteria orienteering problem. Note that the above model differs from more traditional models of orienteering problems in terms of both having distance minimization as a second objective and Equation 1f.
Solving the proposed model with a given set C^d of requesting customers and with their request times t_i^r, results into the Pareto-set of nondominated solutions. This set of solutions may then be compared with the solution obtained by applying decision rules in real-time. In the following Section 4 we propose an algorithmic approach to determining the Pareto-set of nondominated solutions.

4 Evolutionary Multi-objective Approach

Evolutionary Algorithms (EAs) are popular in solving unconstrained multi-objective optimization problems (MOPs) of the form $\min\{f(x) : x \in X\}$ where $f(x) = (f_1(x), \ldots, f_d(x))'$ is a vector-valued mapping with $d \geq 2$ objective functions $f_i : X \to \mathbb{R}$ for $i = 1, \ldots, d$ that are to be minimized simultaneously. The optimality of a MOP is defined by the concept of *dominance*.
Let $u, v \in F \subseteq \mathbb{R}^d$ where F is equipped with the partial order $\preceq$ defined by $u \preceq v \Leftrightarrow \forall i = 1, \ldots, d : u_i \leq v_i$. If $u \prec v \Leftrightarrow u \preceq v \wedge u \neq v$ then v is said to be *dominated by u*. An element u is termed *nondominated* relative to $V \subseteq F$ if there is no $v \in V$ that dominates u. The set $\mathsf{ND}(V, \preceq) = \{u \in V \mid \nexists v \in V : v \prec u\}$ is called the *nondominated set* relative to V.
If $F = f(X)$ is the objective space of some MOP with decision space X and objective function $f(\cdot)$ then the set $F^* = \mathsf{ND}(f(X), \preceq)$ is called the *Pareto-front* (PF). Elements $x \in X$ with $f(x) \in F^*$ are termed *Pareto-optimal* and the set X^* of all Pareto-optimal points is called the *Pareto-set* (PS). Moreover, for some $X \subseteq \mathbb{R}^n$ and $f : X \to \mathbb{R}^d$ the set $\mathsf{ND}_f(X, \preceq) = \{x \in X : f(x) \in \mathsf{ND}(f(X), \preceq)\}$ contains those elements from X whose images are nondominated in image space $f(X) = \{f(x) : x \in X\} \subseteq \mathbb{R}^d$.

4.1 Algorithmic framework of NSGA2

In general, EAs initialize a set of solutions (i.e., the population), evaluate them by the objective function, select solutions for generating new solutions by random variation, and decide after evaluation of the new solutions which solutions should be selected from old and new solutions to form the population of the next iteration.
In the multi-objective setting the selection from old and new solutions needs special measures since it is not guaranteed that all pairs of solutions are mutually comparable. Therefore evolutionary multi-objective EAs frequently deploy a two-stage selection method.
A population P can be partitioned in h disjunct nondominated sets $R_1, \ldots, R_h$ where h is the height of the partially ordered set P:

$$R_1 = \mathsf{ND}_f(P, \preceq) \ \text{ and } \ R_k = \mathsf{ND}_f\left(P \setminus \bigcup_{i=1}^{k-1} R_i, \preceq\right) \text{ for } k = 2, \ldots, h \text{ if } h \geq 2.$$

This procedure is known by the term *nondominated sorting* (Goldberg, 1989). Evidently, every element from R_j is dominated by some individual in R_i if $i < j$.
The *crowding distance* $d_c(y, R_k)$ of some objective vector $y \in R_k$ is intended for reflecting the density of points around $y \in R_k$ (Deb et al., 2002) and may be formalized as follows:

Definition 1 *Let $L_\alpha(A) = \{a \in A : a < \alpha\}$ and $U_\alpha(A) = \{a \in A : a > \alpha\}$ denote subsets of $V \subseteq \mathbb{R}$ with values lower resp. upper than threshold $\alpha \in \mathbb{R}$. Suppose that $e_i \in \mathbb{R}^d$ is the ith unit vector and $\pi_i(Y) = \{v \in \mathbb{R} : v = y'e_i \text{ with } y \in Y\}$ is the projection of set $Y \subseteq \mathbb{R}^d$ to dimension $i = 1, \ldots, d$. The value*

$$d_c(y, Y) = \sum_{i=1}^{d} \frac{d(y_i, L_{y_i}(\pi_i(Y))) + d(y_i, U_{y_i}(\pi_i(Y)))}{\max\{v : v \in \pi_i(Y)\} - \min\{v : v \in \pi_i(Y)\}}$$

is termed the crowding distance *of y in Y if there exists no $i \in \{1,\ldots,d\}$ with $L_{y_i}(\pi_i(Y)) \cap U_{y_i}(\pi_i(Y)) = \emptyset$; otherwise $d_c(y,Y) = \infty$.* □

Algorithm 1 Pseudo code of the NSGA2.

```
1: draw multiset $P$ with $\mu$ elements $\in X$ at random
2: repeat
3:    generate $\mu$ offspring $x_1,\ldots,x_\mu \in X$ from $P$ by variation
4:    $P = P \cup \{x_1,\ldots,x_\mu\}$
5:    build ranking $R_1,\ldots,R_h$ from $P$
6:    set $P = \emptyset$ and $i = 1$
7:    while $|P| + |R_i| \leq \mu$ do
8:       $P = P \cup R_i$ ; $i = i + 1$
9:    end while
10:   $k = \mu - |P|$
11:   if $k > 0$ then
12:      remove those $|R_i| - k$ elements from $R_i$ that have smallest crowding distance
13:   end if
14:   $P = P \cup R_i$
15: until stopping criterion fulfilled
```

The algorithm starts by generating an initial parental population consisting of μ (usually) random solutions (1). Thereafter, it continues in an evolutionary loop which generates μ offspring solutions by variation (3), then ranks the union set of parents and offspring regarding nondomination (5), and finally creates a new parental population by adding better ranked solutions until it contains μ or more individuals (8). In case the new population size is larger than μ, population size is reduced to μ by removing solutions with worst rank and smallest crowding distance (12). In order to adapt the general algorithmic concept of NSGA2 to our problem class, we had to specify an encoding of solutions and suitable variation operators.

4.2 Encoding of solutions

A solution of the given problem as defined in Section 3 has multiple components: on the one hand a subset of customers from C has to be selected, while on the other hand an optimal tour for visiting those customers must be determined. Thus, the encoding of a solution candidate consists of a permutation string and a binary string both of length $N-2$. The permutation string encodes the sequence of visits to all customers, while the binary string $B = (b_2,\ldots,b_{N-1}) \in \{0,1\}^{N-2}$ indicates for each customer $i \in C \setminus \{1,N\}$ whether it is visited ($b_i = 1$) or not ($b_i = 0$). Note that this encoding disregards the start ($i = 1$) and end ($i = N$) depots, as they are always part of the tour. Finally, to distinguish between the subsets C^a and C^d of C, we introduce a probability encoding for each position in the binary string as strategy parameters. Thereby, we can assign a change probability $p_i \in [0,1]$ for each element in the binary string. If $p_i = 0$, the value of b_i will never change throughout the variation process. This way we can fix customers from C^a as obligatory customers. A probability $0 < p_i \leq 1$, however, indicates a certain variability of value b_i, allowing us to denote dynamic customers from C^d. Fig. 1 schematically depicts this encoding.
The encoding scheme allows a direct interpretation: An actual tour can be constructed from the permutation string by only considering the active customers. Fig. 2 shows three solutions for a problem with eight customers (two of them are the start and end depots and excluded from the encoding) and different activations. In the left scenario only customers 3 and 4 are considered as active resulting in a tour that only visits customer 3 after customer 4. In the middle scenario customers 3, 4, and 5 are active. The resulting tour from the permutation considers, with sequence $5 \to 3 \to 4$, only these active customers. All other

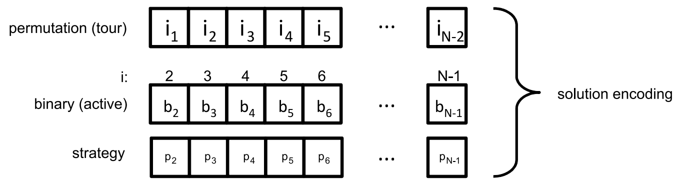


Figure 1: Encoding of a single solution in NSGA2: a permutation chromosome encodes the tour, a binary string encodes whether a customer is active in this tour, and for each gene, a strategy parameter denotes the variability in the binary string.

permutation entries are ignored. In the right scenario, all customers are active and consequently also part of the tour.

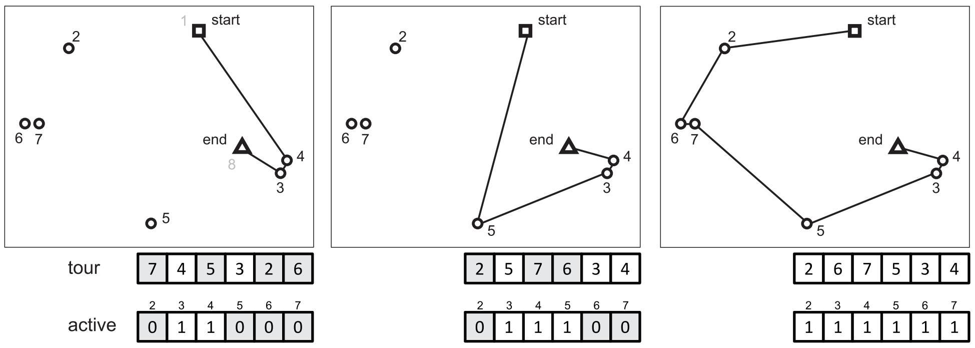


Figure 2: Exemplary encoding of three (Pareto-optimal) solutions for a problem with six customers, two of those active in the leftmost case, three active in the middle and all active in the right case.

Request times t_j for a customer $j \in C$ can easily be integrated into the cost computation for traveling an edge (i, j) during evaluation. Note that strategy parameters p_i are ignored during evaluation as they only concern variation.

4.3 Variation operators and local search

Offspring generation in evolutionary algorithms implies the application of variation operators, specifically mutation and crossover (Algorithm 1, line 3). For the proposed encoding of the considered problem we design specific operators, which are combined from common operators and briefly described in the following.

4.3.1 Swap/Flip-Mutation

Changes to the permutation string are done by repeated pairwise swapping of randomly selected entries and controlled by a parameter $\sigma_p \in \mathbb{N}$ which determines the maximum number of swaps performed. For each application of the operator the number of swaps is drawn uniformly random from $\{0, \ldots, \sigma_p\} \subset \mathbb{N}$. At the same time, the binary string undergoes a flip mutation, where each position $b_i \in \{0, 1\}$ in the string is flipped with probability $p_i \in [0, 1]$ taken from the strategy parameters. Obviously, if $p_i = 0$, mutation is deactivated for the specific entry b_i. This enables us to mask certain customers as obligatory and exclude them from variation.

4.3.2 PMX/One-Point-Crossover

The crossover of two solutions in order to produce new offspring is performed with Partially Mapped Crossover (PMX) for the permutation string and with One-Point-Crossover for the binary string. In PMX offspring is generated by interchanging a sub-sequence of the parental solutions and repairing violations to the permutation (due to double entries) by a mapping procedure, for details see Michalewicz (1999). One-Point-Crossover recombines offspring by splitting up the binary string at a random position and alternately connecting the parts. Note that crossover does not interfere with the activation of customers.

4.3.3 2-OPT Local Search

In addition to the above described variation operators we integrated optional local search—especially for optimizing tour length—at two positions in Algorithm 1. Instead of starting with an arbitrary set of initial solutions (and thus with arbitrary tours), 2-OPT local search can be activated to optimize the initial population (line 1). Further, 2-OPT local search can be activated after variation (line 3).
2-OPT local search is a simple but widely accepted mechanism to optimize the tour length by removing edge crossings from a tour. It is based on the insight, that in euclidean graphs planar tours are more cost-efficient than non-planar. In our implementation we focused on the steepest descent variant of 2-OPT, which selects the most effective interchange of edges per optimization step. The number of steps for 2-OPT is controllable via the parameter r_{2OPT}.

5 Dynamic Approaches

The problem described in Section 3 may be considered as the a-posteriori problem that can be formulated as soon as the vehicle has reached the end depot. A decision maker will be able to use the solutions determined by the multi-objective approach of Section 4 for analyzing the quality of the decisions he made for routing the vehicle from the start depot to the end depot. These routing decisions are in practice typically made in real-time, i.e., they are made dynamically while the vehicle is operated.
One common approach to dynamic vehicle routing is to revise the current routing plan each time a new customer request occurs, i.e., at each point in time denoted as t_i in the model of Section 3. Over the past decade a variety of approaches to making such plan revisions have been proposed in the dynamic vehicle routing literature. In particular, a number of authors (e.g., Meisel, 2011; Thomas, 2007) have recently considered waiting strategies for making plan revisions with dynamic orienteering problems.
Along the lines of the observations of Feillet et al. (2005), waiting strategies require the bi-objective problem to be handled as a single-objective problem with additional constraints. In order to be able to apply a waiting strategy the decision maker needs to select an upper bound for the total travel time the vehicle is allowed to take for moving from the start depot to the end depot. Applying the waiting strategy then implies repeated decisions about whether or not the vehicle should spend idle time at certain locations in the service region. By allocating idle time the decision maker aims at efficiency gains due to the occurrence of close-by customer requests.
We consider three basic waiting strategies for analysis with the approach of Section 4. The three strategies have in common that

- a maximum amount of time available for routing the vehicle must be selected in advance,
- a decision about confirmation or rejection is made each time a new customer request occurs,
- a new request is confirmed only if both the request and all previously confirmed requests can be served within the remaining amount of time,
- a waiting decision is made if the vehicle has just arrived at a customer location, or if the vehicle currently waits and a new request appears.

However, the strategies are different with respect to the principle according to which waiting time is allocated. We consider the following three principles:

- Wait First (WF): The vehicle waits at the start depot and continues confirming customer requests until the length of the planned route equals the remaining time.

- Drive First (DF): The vehicle only waits at its current customer location if both waiting time is available and the planned route only contains the end depot.
- Distributed Waiting (DW): The total amount of available waiting time is distributed equally among all customer locations of the current planned route. The amount of time to be spent at customer locations is recalculated as soon as a new customer request is confirmed.

Each of these three principles implies its own type of routing behavior. Due to the stochastic nature of customer requests any of the resulting waiting strategies may be the best performing one in a particular real-time scenario. Deriving general conclusions about which strategy to apply in such a scenario is a difficult task already in the presence of only the single objective of maximizing the number of customers served (e.g., Meisel, 2011; Thomas, 2007). Deliberate selection of a well-performing dynamic approach becomes even harder if the second objective of minimizing total travel time has to be considered additionally.

6 Experimental setup

6.1 Test problems

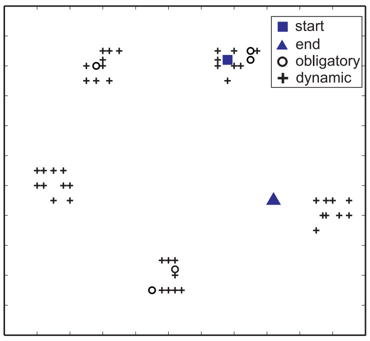
We designed our test problems for evaluation based on two representative geographies. In the first geography, customers are grouped into distinct clusters, while in the second geography, customers are uniformly distributed in euclidean space. These scenarios represent two pure stereotypes of geographies from which any other geography can be constructed. For each geography, we created two instances with 5 obligatory and 44 dynamic (*few*) as well as 44 obligatory and 5 dynamic (*many*) customers respectively. These instances are shown in Fig. 3a-d. We denote the clustered geography with CL and the uniform geography with U. Request times for all dynamic customers are generated via a Poisson process. In case of 44 dynamic customers we use an inter-arrival time of 10 time units. For 5 dynamic customers, we use inter-arrival times of 90 time units.

6.2 Performance measurement

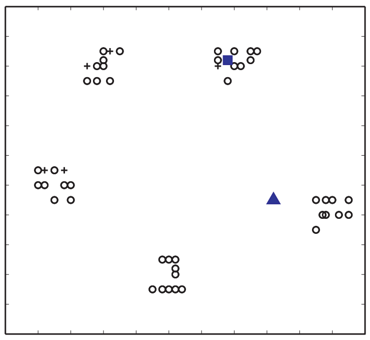
For evaluation of algorithm performance and comparison of solution quality in the multi-objective context, we apply two central methods from experimental multi-objective performance assessment, namely the Hypervolume (Zitzler et al., 2003) and the empirical attainment function (Fonseca and Fleming, 1996).
The *Hypervolume* metric computes the normalized volume enclosed by a nondominated solution set in objective space and a given reference point R along the coordinate axes, see Fig. 4a. If the reference point is chosen as always dominated point in objective space, a larger hypervolume value indicates a better approximation of the true Pareto-front with respect to convergence and spread of found solutions. This indicator can be used to judge on the approximation quality as well as a scalar value which can be traced over generations to show convergence behavior of a multi-objective algorithm.
The *Empirical Attainment Function* allows the statistical interpretation of stochastic multi-objective algorithms by mapping the results of repeated algorithm runs to so called $k\%$-attainment surfaces. Each $k\%$-surface partitions the objective space in two areas: the area which is dominated by $k\%$ solution sets over all runs and the area, which is not dominated, see Fig. 4b. Consequently, the best solutions obtained over all runs form the 0%-surface, while the worst solutions from all runs form the 100%-surface. In our evaluation we always also provide the 50%-surface as median.

6.3 Algorithmic settings

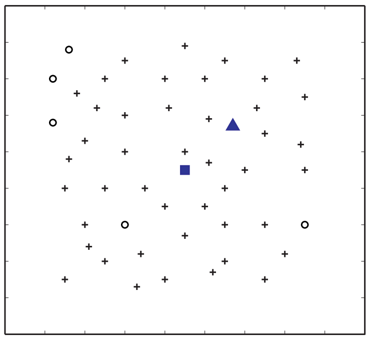
NSGA2 was applied with the before described variation operators and 2-OPT local search. For the Swap/Flip-mutation, $\sigma_p = 2$ was chosen, as this setting yielded best behavior in a tested range of $\sigma_p \in \{1,\dots,5\}$ for the considered test instances. Further, strategy parameters $p_i, i \in C^d$ were initialized with $\frac{1}{N-2}$ for all dynamic customers, depending on the considered instance. The population size was set



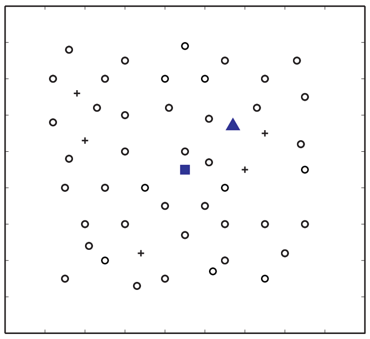
(a) Clustered (CL), 5 customers obligatory (few).



(b) Clustered (CL), 44 customers obligatory (many).

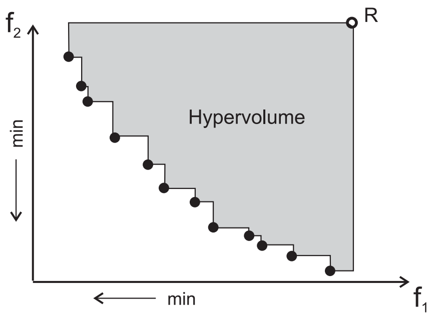


(c) Uniform (U), 5 customers obligatory (few).

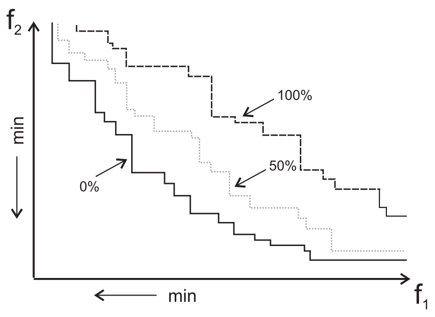


(d) Uniform (U), 44 customers obligatory (many).

Figure 3: Depiction of the test instances used during experiments: Clustered and uniform geographies with 44 (many) and 5 (few) obligatory customers.



(a) Hypervolume.



(b) Emp. Attainment Function

Figure 4: Schematic depiction of the Hypervolume indicator (left) and the empirical attainment function (right) for bi-objective problems.

to 100 individuals and experiments were repeated 20 times with independent random initialization. As termination criterion, we chose absolute run times from $t = \{60s, 120s, 240s\}$ to ensure comparability with the applied dynamic strategies executed in CPLEX. Finally, all experiments were performed with and without 2-Opt local search, to analyze the benefit of applying a specific and problem-related local search method. In the local search cases, 2-OPT was applied to the initial population and after variation in NSGA2. For the initial population, $r_{2OPT,init} = 100$ optimization rounds were performed for each individual, whereas 2-Opt was only activated every 100 generations and for only two optimization rounds after variation ($r_{2OPT,algo} = 100$). The algorithm as well as the 2-Opt local search were implemented in

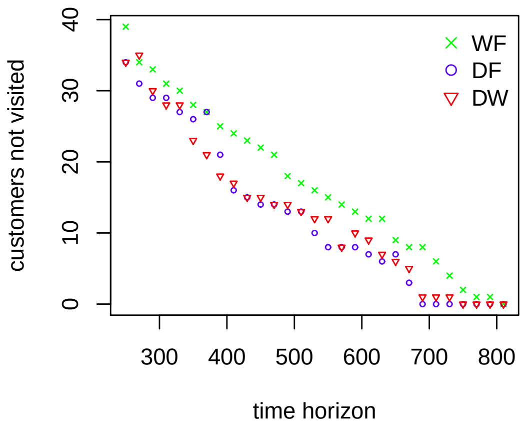
the jMetal Framework by Durillo and Nebro (2011) and executed on a Intel i7-3667U CPU machine at 2.5GHz with 8GB RAM.
In order to be able to analyze the performances of the dynamic waiting strategies with respect to the results of the multi-objective approach, we applied each strategy to each problem instance with a variety of predefined time horizons. For each instance we first applied each strategy with a very short time horizon, and then increased the time horizon in increments of twenty time units up to the value at which all waiting strategies are able to confirm all dynamic customers. All optimization problems that occur during execution of the waiting strategies are solved by the CPLEX solver.

7 Computational Results

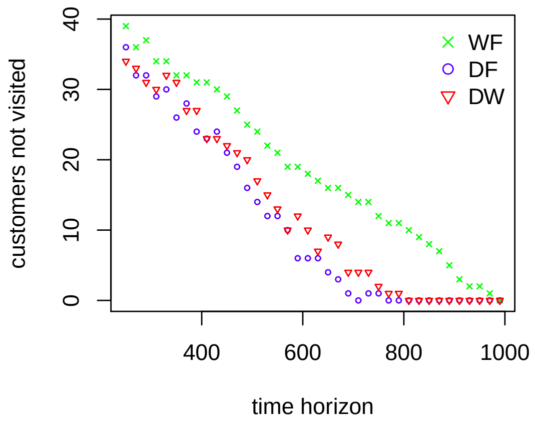
Section 7.1 gives an overview of how the strategies of Section 5 perform when applied to the problems of Section 6.1. Section 7.2 compares the results of Section 7.1 with the bi-objective approach of Section 4.

7.1 Performances of the Dynamic Approaches

For the two instances with few obligatory customers the numbers of non-visited customers are displayed in Fig. 5. With both of these instances, WF never performs best. In case of uniformly distributed customer locations (Subfig. 5b) DW sometimes outperforms DF if the time horizon is relatively short, but never outperforms DF if the time horizon is above a certain threshold. No obvious pattern of the relative performances of DW and DF can be identified with clustered locations (Subfig. 5a), which indicates that the combination of request times and of added structure in terms of clusters results into particularly hard problem instances. The two problem instances with many obligatory customers showed similar structure as the two instances with few obligatory customers.



(a) instance **few, CL**



(b) instance **few, U**

Figure 5: Comparisons of the performances of dynamic approaches at different time horizons.

7.2 Evolutionary Algorithm results and comparison with best Dynamic Approach

The evaluation of our proposed methodology is twofold: we first discuss the performance of the NSGA2 approach regarding its ability to solve the considered vehicle routing problem with time windows. Then, we compare the dynamic approaches to the evolutionary determined results.
The performance of the evolutionary multi-objective algorithm is analyzed by means of the hypervolume indicator. Fig. 6 shows the behavior of the HV distribution in the course of the generations based on 20 individual runs of 120s. It becomes obvious that the algorithm is able to improve initial solution quality

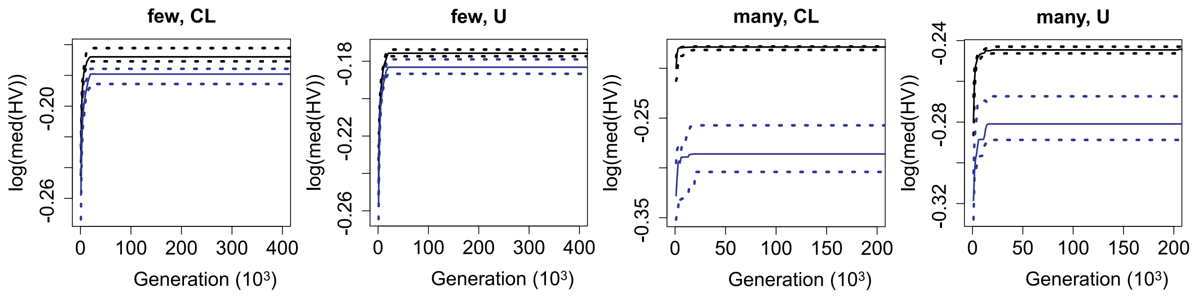


Figure 6: HV (lower quartile, median, upper quartile) in the course of the generations for the different problems after 120s. Runs with (black) and without (blue) local search are compared. Figures are restricted to the first $200 \cdot 10^3$ resp. $400 \cdot 10^3$ generations. $R = (1200, 50)$; lower normalization bounds are (0;0) for few and (0;40) for many obligatory customers.

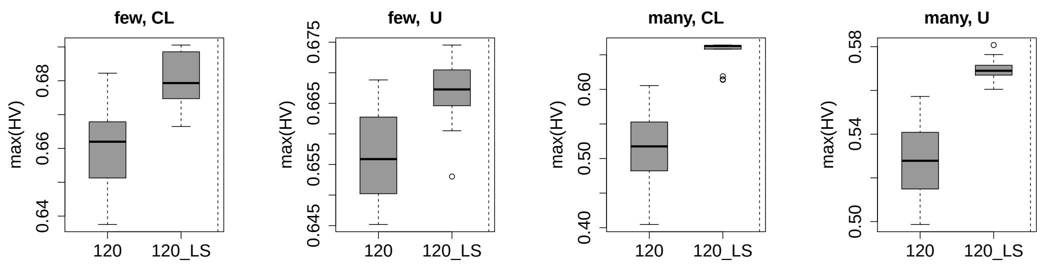


Figure 7: Boxplots of maximum hypervolume for all 20 runs after 120s; test problem few (two leftmost) and many (two rightmost)

very fast, and after a moderate number of generations almost a stagnating behavior regarding HV can be observed. Furthermore, the clear superiority of integrating local search is reflected. The latter observation is also stressed by Fig. 7 in which the maximum HV distribution of all runs is summarized by boxplots on all instances. The local search based variants (denoted as 120_LS) result in significantly higher HV values than the variants operating without local searches (denoted as 120).

The empirical attainment surface plots in Fig. 8 show approximated Pareto-fronts of the problems and indicate that the surface distribution is right skewed such that the best and median surface are closer together than the median and worst ones. Furthermore, the EA successfully generates nondominated solutions covering the whole range of both objectives.

If compared to the dynamic approaches, hardly any of the solutions generated by these approaches dominates the EA solutions. A detailed analysis of the top subfigures of Fig. 8 show, that the evolutionary strategy is superior to dynamic approaches especially for instances with few obligatory customers. The more customers have to be served, the better dynamic strategies seem to perform. This behavior, however, is due to the different focus of the EA and the dynamic strategies: While the dynamic strategies aim for serving most customers in a given *single* time window, the evolutionary algorithm strives for generating the (almost) Pareto-optimal solution set in a single run. Thus, the EA optimizes tour length for *all* solutions simultaneously, where optimal tours with many obligatory customers are more difficult to find than tours with less obligatory customers. The lower subfigures of Fig. 8 show that the EA is able to outperform the dynamic strategies also for many obligatory customers, if the problem is restricted to these cases.

8 Conclusions

In this paper, we addressed the routing of a vehicle with time windows for the first time with the well established multi-objective evolutionary approach NSGA2 and compared it in the light of decision

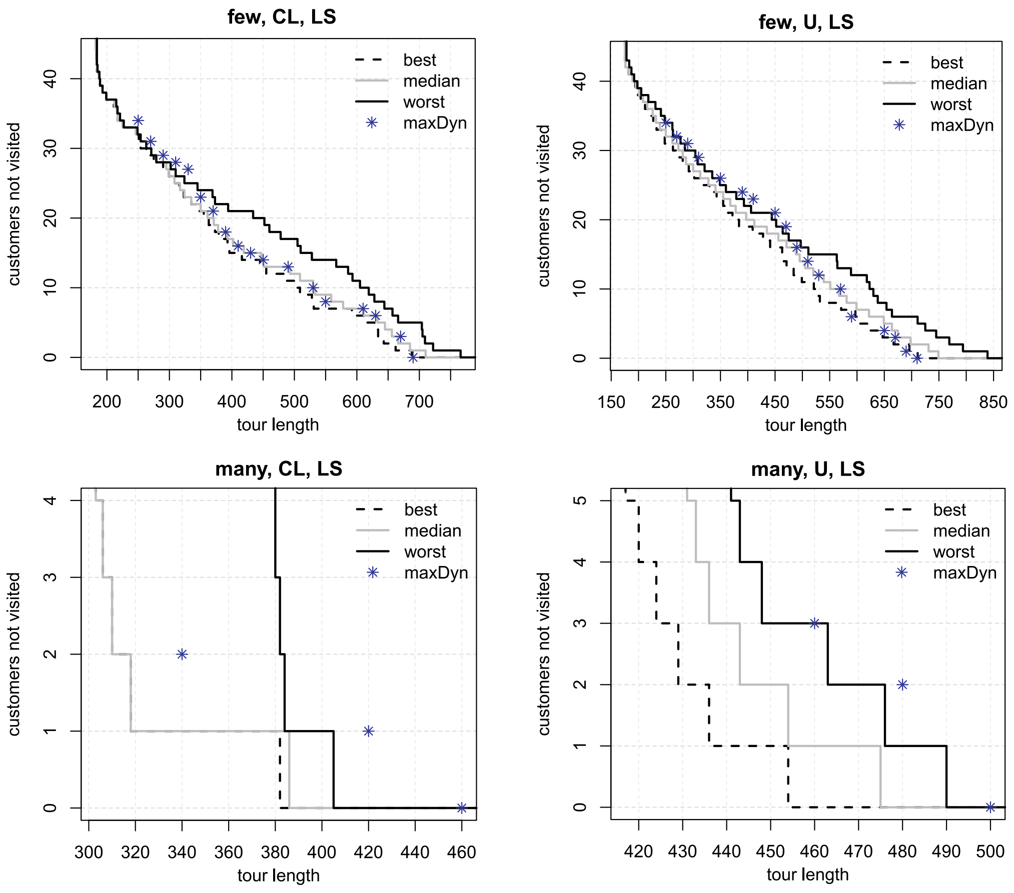


Figure 8: Empirical Attainment surfaces of the local search variants for all problem instances after 120s. The best solutions of the dynamic approaches (maxDyn) are visualized by blue stars.

support with current single-objective dynamic strategies.The rational of this methodology is to provide an a-posteriori assessment tool for improving strategy selection for real-time routing decision makers. Therefore, NSGA2 was adapted in encoding and variation operators. Additionally, extensive experiments were conducted, that show the good performance of the evolutionary multi-objective approach and the high quality of the dynamic strategies available. Overall, we show that there is huge potential in considering the proposed multi-objective strategy as assessment tool. Even with standard parametrization, basic operators, and a minimum of computational effort, the evolutionary approach is able to rapidly find a good near-optimal solution set for all trade-offs in a single run. This finally allows evaluation of dynamic strategies regarding overall reachable trade-off solutions and enables decision makers to decide on suitable dynamic strategies for vehicle tour planning.

Future work regarding the evolutionary approach will follow the lines of Rudolph et al. (2014) who have generalized the concept of the reference point method (Wierzbicki, 1980) to a method that aims at finding solutions on the Pareto-front that are closest to a *set* of user-defined reference points instead of a single point. The focus on this part of the Pareto-front that is closest to the so-called *aspiration set* will save significant computational effort and certainly fits the idea of considering time windows for tour planning. Moreover, this approach might also be extended to operate in interactive mode in which the decision maker has the opportunity to change the location of the aspiration set in the course of the run and may pave the road to dynamic multi-objective approaches to solve the routing of a vehicle with time windows. Needless to say, such an approach is realizable easily up to three or more objectives but raises a number of questions in case of more than three objectives. These questions are related to visualization of high-dimensional data and are an active field of research.

References

Berube, J.-F. et al. (2009). "An exact [epsilon]-constraint method for bi-objective combinatorial optimization problems: Application to the Traveling Salesman Problem with Profits." *European Journal of Operational Research* 194 (1), 39–50.

Deb, K. et al. (2002). "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II." *IEEE Transactions on Evolutionary Computation* 6 (2), 182–197.

Dell'Amico, M. et al. (1995). "On prize-collecting tours and the asymmetric travelling salesman problem." *International Transactions in Operational Research* 2 (3), 297–308.

Durillo, J. and A. Nebro (2011). "jMetal: A Java framework for multi-objective optimization." *Advances in Engineering Software* 42, 760–771.

Feillet, D. et al. (2005). "Traveling Salesman Problems with Profits." *Transportation Science* 39 (2), 188–205.

Filippi, C. and E. Stevanato (2013). "Approximation schemes for bi-objective combinatorial optimization and their application to the TSP with profits." *Computers & Operations Research* 40 (10), 2418–2428.

Fonseca, C. and P. Fleming (1996). "On the performance assessment and comparison of stochastic multiobjective optimizers." In: *Parallel Problem Solving from Nature, PPSN IV*. Ed. by H.-M. Voigt et al. Vol. 1141. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 584–593. ISBN: 978-3-540-61723-5.

Gendreau, M. et al. (1998). "A branch-and-cut algorithm for the undirected selective traveling salesman problem." *Networks* 32 (4), 263–273.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston: Addison-Wesley.

Golden, B. L. et al. (1987). "The orienteering problem." *Naval Research Logistics* 34 (3), 307–318.

Jozefowiez, N. et al. (2008). "Multi-objective Meta-heuristics for the Traveling Salesman Problem with profits." *Journal of Mathematical Modelling and Algorithms* 7 (2), 177–195.

Kantor, M. G. and M. B. Rosenwein (1992). "The Orienteering Problem with Time Windows." *The Journal of the Operational Research Society* 43 (6), 629–635.

Keller, C. P. and M. Goodchild (1988). "The multiobjective vending problem: A generalization of the traveling salesman problem." *Environ. Planning B: Planning Design* 15 (2), 447–460.

Laporte, G. and S. Martello (1990). "The selective travelling salesman problem." *Discrete Applied Mathematics* 26 (2-3), 193–207.

Meisel, S. (2011). *Anticipatory Optimization for Dynamic Decision Making*. Vol. 51. Operations Research/Computer Science Interfaces Series. Springer New York.

Michalewicz, Z. (1999). *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd Edition. Berlin: Springer.

Rudolph, G. et al. (2014). "A Multiobjective Evolutionary Algorithm Guided by Averaged Hausdorff Distance to Aspiration Sets." In: *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*. Ed. by A.-A. Tantar et al. Springer, pp. 261–273.

Thomas, B. (2007). "Waiting Strategies for Anticipating Service Requests from Known Customer Locations." *Transportation Science* 41 (3), 319–331.

Wierzbicki, A. (1980). "The use of reference objectives in multiobjective optimization." In: *Multiple Objective Decision Making, Theory and Application*. Ed. by G. Fandel and T. Gal. Springer, pp. 468–486.

Zitzler, E. et al. (2003). "Performance assessment of multiobjective optimizers: An analysis and review." *IEEE Transactions on Evolutionary Computation* 7 (2), 117–132.