Lecture Notes in Economics and Mathematical Systems 624

Martin Josef Geiger • Walter Habenicht
Marc Sevaux • Kenneth Sörensen
Editors

Metaheuristics in the Service Industry

Springer

Dr. Martin Josef Geiger
University of Southern Denmark
Department of Business and Economics
Campusvej 55
5230 Odense
Denmark
mjg@sam.sdu.dk

Prof. Dr. Marc Sevaux
Université de Bretagne-Sud
Lab-STICC, CNRS-UEB
2 rue de Saint Maudé - BP 92116
56321 Lorient - Cedex
France
marc.sevaux@univ-ubs.fr

Prof. Dr. Walter Habenicht
University of Hohenheim
Production and Logistics Management
Department
Schloss, Osthof-Nord
70593 Stuttgart
Germany
habenicht@uni-hohenheim.de

Dr. Kenneth Sörensen
University of Antwerp
Faculty of Applied Economics
Prinsstraat 13
2000 Antwerp
Belgium
kenneth.sorensen@ua.ac.be

ISSN 0075-8442
ISBN 978-3-642-00938-9 e-ISBN 978-3-642-00939-6
DOI 10.1007/978-3-642-00939-6
Springer Dordrecht Heidelberg London New York

Library of Congress Control Number: 2009926020

Cover design: SPi Publisher Services

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

While the cost- and material-effective production of tangible goods has been the emphasis of early industrialization, developed countries show an increasing importance of services. Similar to situations arising from manufacturing, optimization problems can be identified here that can be addressed using modern metaheuristic approaches.

The book presents a collection of articles describing recent advances in metaheuristics, in particular with applications in the fast developing service industry. The compilation of these papers was preceded by the 8th meeting of the EU/ME working group, held in October 2007 in Stuttgart, Germany. EU/ME, the *European Chapter on Metaheuristics* is a working group within EURO, the Association of European Operational Research Societies.

While some of the results given in this volume have been presented and discussed during the 2007 workshop, a wider call for papers followed that invited all fellow researchers to contribute to the book. Each article has been peer-reviewed by several referees, and as a result a subset of all received articles has been accepted for publication.

With regard to the scope of the book, applications in areas of modern services are targeted:

- Transportation and logistics play an important role here. A bicriterion traveling salesman problem is tackled in the article of Schmitz and Niemann, and an application of the vehicle routing problem is studied by Rieck and Zimmermann. Toll pricing in road networks is investigated by Dimitriou and Tsekeris, while the article of Ortega-Mier, Delgado Hipólito, and García-Sánchez solves the problem of locating a treatment plant in a reverse logistics network. Vansteenwegen, Souffriau, Vanden Berghe, and Van Oudheusden contribute with two articles. The first describes the interesting application of tourist trip planning, while the other is dedicated to the crane operations in train terminals.
- Besides classical logistical problems, areas such as production scheduling and multi-item economic order quantity problems are addressed. The former problem is studied by Czogalla and Fink, while for the latter a contribution is made by Baykasoğlu and Göçken.
- Moreover, a financial application is considered in the form of a index tracking problem by the work of di Tollo and Maringer.

The production of this volume would not have been possible without the support of numerous colleagues. We owe our thanks to the referees, who here have to remain anonymous. Explicitly to mention are, however, the sponsors of the 8th EU/MEeting that contributed with their financial aid to the success of the event:

- The Association of European Operational Research Societies EURO
- The "Universitätsbund Hohenheim e. V."
- The German Society of Operations Research GOR e. V.
- The "Forschungszentrum Innovation und Dienstleistung FZID," the research center on innovation and services of the University of Hohenheim

Odense	*Martin Josef Geiger*
Stuttgart	*Walter Habenicht*
Lorient	*Marc Sevaux*
Antwerp	*Kenneth Sörensen*
February 2009	

Contents

Contributors

Adil Baykasoğlu University of Gaziantep, Department of Industrial Engineering, Gaziantep, Turkey, baykasoglu@gantep.edu.tr

Jens Czogalla Helmut-Schmidt-University/UniBw Hamburg, Holstenhofweg 85, 22043 Hamburg, Germany, czogalla@hsu-hamburg.de

Joaquín Delgado Hipólito Technical University of Madrid, José Gutiérrez Abascal 2 (Madrid) 28006, Spain, joaquin.delgado@upm.es

Giacomo di Tollo Dipartimento di Scienze, Università "G. D'Annunzio" Chieti–Pescara, Viale Pindaro 42 - Pescara, Italia, ditollo@sci.unich.it

Loukas Dimitriou Department of Transportation Planning and Engineering, School of Civil Engineering, National Technical University of Athens, Iroon Polytechniou 5, 15773 Athens, Greece, lucdimit@central.ntua.gr

Andreas Fink Helmut-Schmidt-University/UniBw Hamburg, Holstenhofweg 85, 22043 Hamburg, Germany, andreas.fink@hsu-hamburg.de

Álvaro García-Sánchez Technical University of Madrid, Jose Gutierrez Abascal 2 (Madrid) 28006, Spain, alvaro.garcia@upm.es

Tolunay Göçken Department of Industrial Engineering, University of Gaziantep, Gaziantep, Turkey, Tolunay.Gocken@gantep.edu.tr

Dietmar Maringer Faculty of Economics and Business Administration, University of Basel, Petersgraben 51, Postfach, 4003 Basel, Switzerland, Dietmar.Maringer@unibas.ch

Sebastian Niemann Fachhochschule Trier, Department of Computer Science, Schneidershof, 54293 Trier, Germany

Míguel Ortega-Mier Technical University of Madrid, José Gutiérrez Abascal 2, Madrid 28006, Spain, miguel.ortega.mier@upm.es

Julia Rieck Clausthal University of Technology, Institute of Management and Economics, Operations Research Group, Julius-Albert-Str. 2, 38678 Clausthal-Zellerfeld, Germany, julia.rieck@tu-clausthal.de

Heinz Schmitz Department of Computer Science, Fachhochschule Trier, Schneidershof, 54293 Trier, Germany, schmitz@informatik.fh-trier.de

Wouter Souffriau Information Technology, KaHo Sint-Lieven, Gebroeders Desmetstraat 1, 9000 Gent, Belgium
and
Katholieke Universiteit Leuven, Centre for Industrial Management, Celestijnenlaan 300A, 3001 Leuven (Heverlee), Belgium, Wouter.Souffriau@kahosl.be

Theodore Tsekeris Centre for Planning and Economic Research (KEPE), Amerikis 11, 10672 Athens, Greece, tsek@kepe.gr

Dirk Van Oudheusden Centre for Industrial Management, Katholieke Universiteit Leuven, Celestijnenlaan 300a - bus 2422, 3001 Leuven, Belgium, Dirk.VanOudheusden@cib.kuleuven.be

Greet Vanden Berghe Information Technology, KaHo Sint-Lieven, Gebroeders Desmetstraat 1, 9000 Gent, Belgium, Greet.VandenBerghe@kahosl.be

Pieter Vansteenwegen Centre for Industrial Management, Katholieke Universiteit Leuven, Celestijnenlaan 300a – bus 2422, 3001 Leuven, Belgium, Pieter.Vansteenwegen@cib.kuleuven.be

Jürgen Zimmermann Operations Research Group, Institute of Management and Economics, Clausthal University of Technology, Julius-Albert-Str. 2, 38678 Clausthal-Zellerfeld, Germany, juergen.zimmermann@tu-clausthal.de

A Bicriteria Traveling Salesman Problem with Sequence Priorities

Heinz Schmitz and Sebastian Niemann

Abstract This paper introduces a bicriteria version of the classical Traveling Salesman Problem (TSP) which is motivated by various applications in the context of service delivery. The additional objective allows to take priorities among locations into account while minimizing the costs of traveling. For this, cities in the input are given in a strict ordering, e.g., due to arrival times of delivery requests. The goal is to compute the set of efficient solutions when both objectives are optimized simultaneously. To the best of our knowledge, this variation of TSP has not been studied before.

After making the notion of priorities precise, we present a local-search algorithm to approximate the set of non-dominated solutions. While still being conceptionally easy, our algorithm employs different means of intensification and diversification in a way we call *breadth-first local search*. We maintain one candidate solution for each possible value of the additional objective in a polynomially-sized archive, and try to improve this set towards the Pareto front. Experimental results with test data from TSPLIB show that this is a reasonable approach to attack the problem.

Keywords Bicriteria traveling salesman problem · Local search · Metaheuristic · Multi-objective discrete optimization · Sequence priorities

1 Introduction

The classical Traveling Salesman Problem (TSP) has numerous real-world applications, for a recent overview we refer to [7]. A typical one is the minimum-cost tour scheduling to fulfill delivery requests from different locations.

Example 1. Let $\{1, \ldots, m\}$ be a set of customer locations such that the earliest request is from location 1, the second earliest from 2 and so on. Obviously, minimizing

H. Schmitz (✉)
Department of Computer Science, Fachhochschule Trier, Schneidershof, 54293 Trier, Germany
e-mail: schmitz@informatik.fh-trier.de

M.J. Geiger et al. (eds.), *Metaheuristics in the Service Industry*, Lecture Notes in Economics and Mathematical Systems 624, DOI 10.1007/978-3-642-00939-6_1,

traveling costs is a reasonable objective for the shipping company. However, if delivery time is crucial, then the tour $(m, m-1, \ldots, 1)$ cannot be considered optimal from a customers perspective since requests are fulfilled in reverse order.

One way around this is to incorporate the *first-come-first-served*-policy. If locations have priorities according to arrival times of delivery requests one can additionally try to maintain this ordering while minimizing traveling costs. Obviously, both objectives are conflicting in general. In such a situation, we would like to present the set of efficient solutions to the decision maker, i.e., the set of all solutions such that there is no other solution having better values in both objectives. To the best of our knowledge, this variation of TSP has not been studied before.

Due to the many applications of TSP there are also other settings where the new objective emerges in a natural way. We have encountered the following situation in a real-world project, where we used the approach presented in this paper.

Example 2. The manager of a single-machine production line wants to schedule tasks such that the overall makespan is minimized. Since there are sequence-dependent setup times this is equivalent to solving TSP instances. On the other hand, there also needs to be achieved some level of service for different sales departments that place orders. To avoid lengthy discussions about what jobs are more important than others, all participants agreed on the *first-come-first-served*-policy. So in fact, the production-line manager needs to solve TSP instances with sequence priorities, i.e., trade-offs between both objectives have to be balanced.

Once priorities are considered in general, they can also be used to implement other preferences, e.g., due to some customer-specific or order-specific properties. As another example we want to mention the agent service brokering problem which can be understood as a generalized TSP problem [2]. Among other criteria, service requests are presented in an ordered manner by client agents. When the service broker tries to choose a set of service providers subject to minimal costs, the broker can additionally take the priorities of services into account.

Our Contribution

Motivated by the above examples we give a formal definition of the *Traveling Salesman Problem with Sequence Priorities* (TSPwSP) which is a straightforward and natural extension of classical TSP (Sect. 2). We observe some easy facts about this bicriteria problem.

Next we design a local-search algorithm called *breadth-first local search* that combines the two typical means of heuristic search, i.e., intensification and diversification, in a novel way (Sect. 3). To intensify the search we explore the neighborhood of an ordered archive of candidate solutions using different neighborhood structures and a variable search-depth (Sect. 3.1). The polynomially-sized archive is such that it keeps one candidate solution for each possible value of the additional objective. A well-balanced amount of diversification is achieved using tabu lists together with a random choice between two problem-specific perturbation operators (Sect. 3.2). Experimental results suggest that this is a promising approach to solve TSPwSP (Sect. 4).

Additionally, our algorithm can be seen as an implementation of a more abstract algorithmic pattern that can be used to solve similar bicriteria problems, namely problems where the range of at least one objective function is polynomially bounded in the input size.

Related Work

Our algorithmic approach is in the same line with other recently published variants of local search, such as Pareto Local Search [12] and Pareto Iterated Local Search [3], but it also differs in a number of aspects. On one hand we also incorporate perturbation operators [10], we make use of multiple neighborhood structures [9], and we keep an archive of candidate solutions. For more details on related local search metaheuristics we refer to [4].

On the other hand we exploit that the range of the additional objective function is polynomially bounded to obtain a *dense* approximation for every such value, and we do not restrict the archive to locally Pareto optimal solutions. Moreover, we combine the mentioned aspects with tabu lists which is a well established method in combinatorial optimization [5]. It has been successfully applied to classical TSP as well as to other multiple-objective problems, e.g., [1, 11]. Other multiple-objective variants of TSP that have been studied in the literature mainly consider multiple cost matrices, e.g., [8].

2 Problem Statement

We define the *Traveling Salesman Problem with Sequence Priorities* (TSPwSP) as follows. As in case of classical TSP an instance $x = (m, C)$ with m cities consists of some cost matrix $C = (c_{i,j})_{m\times m}$ with $c_{i,j} \in \mathbb{N}$. A solution for x is any permutation $t = (a_1, \ldots, a_m)$ of $\{1, \ldots, m\}$ having costs

$$z_1(t) = c_{a_m,a_1} + \sum_{i=1}^{m-1} c_{a_i,a_{i+1}}.$$

Additionally we assume without loss of generality that cities are labeled according to some priority rule, i.e., the city with highest (lowest) priority has label 1 (respectively, m). No extra input data is needed. To measure violations of these priorities we define the penalty resulting from the i-th city on tour t as $p_i = \max\{i - a_i, 0\}$. E.g., if $a_5 = 3$ then city with priority 3 is visited in fifth place and hence $p_5 = 2$. As the second objective function we set

$$z_2(t) = \sum_{i=1}^{m} p_i.$$

Note that this takes the quantity of each penalty into account. In analogy to the tardiness measure in machine scheduling one could also count the number of non-zero penalties or minimize the largest penalty. We do not study these alternatives here.

It is easy to see that $t = (1, 2, \dots, m)$ is the unique tour with $z_2(t) = 0$, and that

$$z_2(m, m-1, \dots, 1) = \begin{cases} (m^2-1)/4, & \text{if } m \text{ is odd,} \\ m^2/4, & \text{otherwise.} \end{cases}$$

Since no solutions with larger z_2-values exist, we have for all tours t that

$$0 \le z_2(t) \le m^2/4. \tag{1}$$

For notational convenience assume that m is even for the remainder of this paper.

We associate with every tour t the vector $z(t) = (z_1(t), z_2(t))$ where both components need to be minimized. As is common in multicriteria optimization we say a tour t *dominates* t' if $z_1(t) \le z_1(t')$, $z_2(t) \le z_2(t')$ and $z(t) \ne z(t')$. If there is no t that dominates t' we call t' *Pareto-optimal* or *efficient*. Observe from (1) that for each input x the number of Pareto-optimal solutions t with pairwise different vectors $z(t)$ is polynomially bounded in the length of x. The optimization goal for instances of TSPwSP is to compute such a set of Pareto-optimal tours.

It is easy to see that TSPwSP is not a special case of bicriteria TSP where a second cost matrix is given. Just note that usually $z_2(t) \ne z_2(t')$ if t' is a cyclic shift of the permutation t. There are also some similarities to the single-machine scheduling problem $1|s_{fg}|\#(C_{\max}, \sum T_j)$ with sequence-dependent setup times where the overall makespan and the total tardiness both need to be minimized (for standard notations for scheduling problems see, e.g., [15]). However, it is not clear how a reduction from TSPwSP to this problem can be achieved such that the quality of solutions is preserved.

3 Algorithm Design

We describe the main design ideas of our algorithm. The overall structure is rather simple: We keep an archive A of candidate solutions, and try to improve its quality via alternation of intensifying and diversifying phases during the search. To organize this, we instantiate the pool template [6, 16] (Algorithm 2) to control the behavior of an iterated local-search procedure (Algorithm 3).

To be more precise, solutions in the polynomially-sized set $A = (t_0, t_1, \dots)$ always have the property that

$$z_2(t_k) = k \quad \text{for } 0 \le k \le m^2/4. \tag{2}$$

Hence we can understand A as a *dense* approximation of the Pareto front since it contains one candidate for each possible value in the range of function z_2. It is not until the final step of the algorithm that a (locally) efficient set of solutions is extracted from the set of best solutions that appeared during the search.

There is a straightforward way to generate a first version of the archive such that (2) holds (Algorithm 1). Starting with $(1, 2, 3, \dots, m)$ we move the first city to the last position in the permutation via successive transpositions that increase the penalty one-by-one. Then we move city 2 in $(2, 3, \dots, m, 1)$ to the second last position resulting in $(3, \dots, m, 2, 1)$. This is repeated until we finally get $(m, \dots, 3, 2, 1)$. Note that not every transposition during this procedure strictly increases the penalty, e.g., when turning $(2, 3, \dots, m, 1)$ into $(3, 2, \dots, m, 1)$.

For $t = (a_1, \dots, a_m)$ denote by $exchng(t, i, j)$ the transposition of a_i and a_j. Then we can state the following algorithm.

Algorithm 1: `init()`

```
begin
    t := (1, 2, ..., m);
    t_0 := t; A := {t_0};
    k := 0;
    for lastpos := m downto 2 do
        for pos := 1 to (lastpos − 1) do
            t := exchng(t, pos, pos + 1);
            if z_2(t) = k + 1 then
                k := k + 1;
                t_k := t; A := A ∪ {t_k}
            end
        end
    end
    return A
end
```

Next we describe what the instantiation of the pool template looks like. One kind of diversification we use is a collection T of tabu lists $T(k)$ for each $0 \le k \le m^2/4$. We take these lists to ensure that a subsequent iteration of the local search yields an archive A' with solutions that all have z_1-values different from the ones of previous iterations. So if $A = (t_0, t_1, \dots)$ is the content of the archive j iterations ago, we let

$$T(k) = (z_1^1, \dots, z_1^l) \quad \text{with } z_1^j = z_1(t_k).$$

By storing values instead of solutions every tabu-list entry excludes numerous other tours. The duration of this effect can immediately be controlled by the length-parameter l. It is one out of just two search parameters, the other one being the number of iterations for the halting condition.

After initializing the archive and the tabu lists, we repeatedly call the local-search procedure `intensify`, we remember the best solutions found so far, update the tabu lists and we apply one out of two randomly-chosen perturbation operators. Inspired by the way intensification is organized (left–right sweeps, see next subsection) we call our approach *breadth-first local search* (BFLS). Together, we have the following algorithmic pattern.

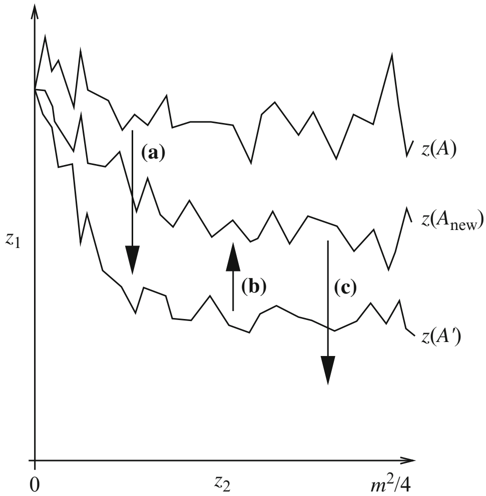


Fig. 1 Every iteration has an `intensify`-step (**a**) followed by the application of a perturbation operator (**b**). Due to forbidden values, the next iteration (**c**) returns a vector $z(A'_{new})$ which is componentwise different from $z(A')$

Algorithm 2: `BFLS`(*maxIterations*, l)

begin
  $A :=$ `init`();
  init T with $T(k) := \emptyset$ for $0 \le k \le m^2/4$;
  repeat
    A':=`intensify` (A, T);
    update $bestArchive$ with A';
    update T with A';
    choose $r \in \{1, 2\}$ randomly;
    $A := \mathcal{O}_r(A')$
  until *maxIterations* reached ;
  return efficient solutions in *bestArchive*
end

A detailed description of the `intensify`-procedure is given in Sect. 3.1 below, while the perturbation operators $\mathcal{O}_1$ and $\mathcal{O}_2$ are explained in Sect. 3.2. They form the second kind of diversification we use in the algorithm.

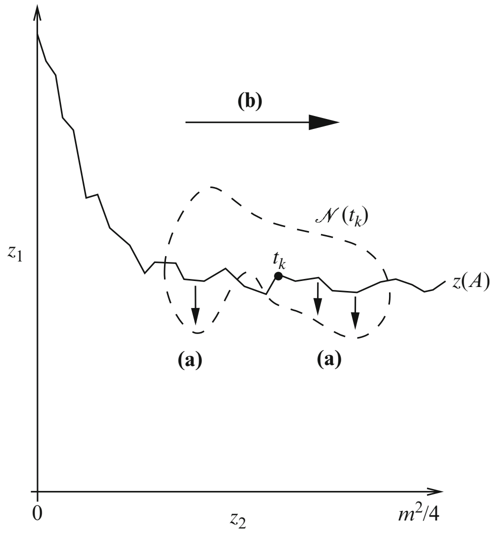
For $A = (t_0, t_1, \ldots)$ denote its coordinates in objective space as $z(A) = (z(t_0), z(t_1), \ldots)$. Then the progress of `BFLS` can be depicted as shown in Fig. 1.

3.1 Intensification

The execution of a single call of the `intensify`-procedure starts with an archive A and performs several left–right sweeps.[1] During each sweep neighborhoods $\mathcal{N}(t_k)$

[1] Note that this intuitive name depends on the order of objectives in the graphical representation we chose to present the Pareto front.

Fig. 2 Archive A is updated (**a**) whenever a non-forbidden but better solution is found in $\mathcal{N}(t_k)$. Each left–right sweep (**b**) repeats this for $k = 0, 1, \ldots, m^2/4$



of $t_k \in A$ for $k = 0, 1, \ldots, m^2/4$ are investigated in this order. Whenever a tour $t \in \mathcal{N}(t_k)$ with $z_2(t) = k'$ is found such that

$$z_1(t) \notin T(k') \quad \text{and} \quad z_1(t) < z_1(t_{k'}) \tag{3}$$

for some $t_{k'} \in A$, then $t_{k'}$ is replaced by t (see Fig. 2).

It has been observed in the literature that the combined use of different neighborhood functions yields better results compared to a single function [3, 9]. We apply two such functions that are efficiently computable but yet effective. The first is simply the exchange neighborhood

$$\mathcal{N}_1(t) = \{exchng(t, i, j) \mid 1 \le i < j \le m\}.$$

For a second one we move every a_i to some position j, i.e., if $t = (\ldots, a_i, \ldots, a_j, \ldots)$ let $move(t, i, j) = (\ldots, a_{i-1}, a_{i+1}, \ldots, a_j, a_i, \ldots)$ and

$$\mathcal{N}_2(t) = \{move(t, i, j) \mid 1 \le i, j \le m\}.$$

The `intensify`-procedure alternates between $\mathcal{N}_1$ and $\mathcal{N}_2$ after every completion of a left–right sweep. This is repeated until A is locally optimal with respect to both neighborhood functions.

Algorithm 3: `intensify`(A,T)

begin
  $s := 1$;
  repeat
    for $k := 0$ **to** $m^2/4$ **do**
      foreach $t \in \mathcal{N}_s(t_k)$ **do**
        $k' := z_2(t)$;
        if (3) holds **then**
          replace $t_{k'}$ by t in A;
        end
      end
    end
    $s := (s \bmod 2) + 1$;
  until A is locally optimal ;
  return A
end

As a result of this procedure we obtain an archive A such that no $t_k \in A$ can be further improved within

$$\mathcal{N}(A) = \bigcup_{t \in A} \bigcup_{s \in \{1,2\}} \mathcal{N}_s(t).$$

So every t_k is not only locally optimal for the value $z_2(t_k)$ within $\mathcal{N}_s(t_k)$, but it is also a best tour for this z_2-value in all other neighborhoods $\mathcal{N}_s(t)$ with $t \in A$ and $s \in \{1,2\}$.

There are two more aspects worth noticing. First, even tours already known to be dominated may contribute with their neighborhood to an improvement of the archive (see again Fig. 2). This is because we do not restrict A to locally Pareto-optimal solutions after each iteration.

Secondly, due to the overlapping of neighborhoods there is a variable search-depth during intensification. If some $t_{k+\delta}$ for $\delta > 0$ is improved to $t'_{k+\delta}$ when looking at $\mathcal{N}_s(t_k)$, the search continues with $\mathcal{N}_s(t'_{k+\delta})$ later during the same sweep. If $t_{k-\delta}$ has changed, then $\mathcal{N}_s(t'_{k-\delta})$ is considered during the next sweep. It turns out that δ can be as large as $m-1$ for both neighborhood functions. To see this let $t = (1,\ldots,m)$ and observe that

$$z_2(t) + (m-1) = z_2(exchng(t,1,m)) = z_2(move(t,1,m)).$$

3.2 Perturbation Operators

When A becomes a locally-optimal fixpoint during intensification, the BFLS algorithm makes a random choice between two perturbation operators in order to diversify the archive while maintaining other (parts of) solutions at the same time.

The first operator simply performs a swap operation by exchanging the first half with the second half of a tour. So if $t = (a_1,\ldots,a_i,a_{i+1},\ldots,a_m)$ with $i = m/2$ then

$$\mathcal{O}_1(t) = (a_{i+1}, \ldots, a_m, a_1, \ldots, a_i).$$

This does not necessarily generate a new solution for *every* z_2-value in the archive, but it can be experimentally observed that a reasonable large fraction of A is rebuilt. Note that this swap operator has the property that penalties usually change but

$$z_1(t) = z_1(\mathcal{O}_1(t)).$$

For our second operator $\mathcal{O}_2$ we would like to obtain just the dual behavior: It should change the z_1-value of a tour $t \in A$ but $z_2(t) = z_2(\mathcal{O}_2(t))$. The design of such an operator is more subtle. To our knowledge, it is not clear how many permutations t' exist with $z_2(t') = z_2(t)$ for some given t, and how they can be computed without exhaustive search. The numbers $S(m, k)$ of permutations t of $\{1, \ldots, m\}$ such that $z_2(t) = k$ are known as integer sequence A062869 from [14].

The idea for $\mathcal{O}_2$ is to define an equivalence relation such that some random $t' = (a'_1, \ldots, a'_m)$ with $z_2(t') = z_2(t)$ can be chosen efficiently from t's equivalence class when $t = (a_1, \ldots, a_m)$ is given. Let

$$P(t) = \{1 \le i \le m \mid p_i = 0\}$$

be the set of positions in t that do not contribute to $z_2(t)$. We say that t' is a *variant* of t, in symbols $t \approx t'$, if and only if for $1 \le i \le m$ it holds that

1. $i \in P(t) \Rightarrow p'_i = 0$
2. $i \notin P(t) \Rightarrow a'_i = a_i$

So t' is obtained from t by permuting elements from $V(t) = \{a_j \,|\, j \in P(t)\}$ without introducing new penalties. It is easy to see that $\approx$ is an equivalence relation and that $t \approx t'$ implies $z_2(t) = z_2(t')$. Observe furthermore that by ii) the part of a tour t that is responsible for t's penalty is carried over to every variant of t.

In general, there are permutations t having $\approx$-equivalence classes of exponential size, e.g., if $t = (m, m-1, \ldots, 1)$ then every permutation of the cities m, $m-1, \ldots, m/2$ yields a variant of t. However, the following producer-consumer type of algorithm efficiently computes $\mathcal{O}_2(t)$ by making a uniform random choice

Algorithm 4: $\mathcal{O}_2(t)$

begin
  $t' := t$;
  $(d_1, \ldots, d_m) := (0, \ldots, 0)$;
  for $i := m$ **downto** 1 **do**
    if $i \in V(t)$ **then** $d_i := 1$ **end**;
    if $i \in P(t)$ **then**
      choose $r \in \{i \le j \le m \mid d_j = 1\}$ randomly;
      $a'_i := r$;
      $d_r := 0$
    end
  end
  return t'
end

from $[t]_\approx$. A 0-1-vector $(d_1, \ldots, d_m)$ stores the candidates from $V(t)$ (producer) that can be put at position $i \in P(t)$ (consumer) as i decreases from m to 1.

It must be noticed that there are permutations with $[t]_\approx = \{t\}$, e.g., if $t = (2, 1, 4, 3, \ldots, m, m-1)$. In such an undesirable case $\mathcal{O}_2$ has no effect. We finally prove in this section how many permutations have single-elemented equivalence classes. To do so, we first show the following characterization.

Lemma 3. *Let* $t = (a_1, \ldots, a_m)$ *be a permutation of* $\{1, \ldots, m\}$. *Then it holds that* $[t]_\approx = \{t\}$ *if and only if* $p_j > 0$ *for all* $i \in P(t)$ *and* j *with* $i < j \leq a_i$.

Proof. We prove both implications by contraposition. So first assume that there is some $i \in P(t)$ and some j with $i < j \leq a_i$ such that $p_j = 0$. If we put a_i in t at position j there is penalty 0 at this position because $a_i \geq j$. On the other hand, we get from $p_j = 0$ that $a_j \geq j > i$. So we can also place a_j at position i in t while having penalty 0 there as well. Since $i \neq j$ the transposition of a_i and a_j yields a strict variant of t.

Conversely, let $t' = (a'_1, \ldots, a'_m) \in [t]_\approx$ with $t' \neq t$. So there must be some $a_i \in V(t)$ and $i, j \in P(t)$ with $a_i = a'_j$ but $i \neq j$. We may assume without loss of generality that $j > i$ since it cannot be the case that $j \leq i$ for all $a_i \in V(t)$. Because $p'_j = 0$ it holds that $a_i = a'_j \geq j$ and with $P(t) = P(t')$ we see that also $p_j = 0$. Together, we identified some $i \in P(t)$ and j with $i < j \leq a_i$ such that $p_j = 0$. □

Next we want to count all permutations with the above property. Assume $P(t) = \{i_1, \ldots, i_k\}$ for some $0 \leq k \leq m$ and $i_1 < \cdots < i_k$. For every $i_l \in P(t)$ it holds that $a_{i_l} \in \{i_l, \ldots, m\}$ and due to the previous lemma we have $i_{l+1} \in \{a_{i_l}, \ldots, m\}$. In order to count these possibilities we consider a tree $B(m)$ with its root labeled 0, and such that every node with label n has successors labeled $n+1, \ldots, m$. Then a node with label n at depth d means $a_{i_d} = n$ and $i_{d+1} = n + 1$, and every leaf of $B(m)$ corresponds uniquely to a permutation t with $[t]_\approx = \{t\}$. An easy induction shows that $B(m)$ has 2^{m-1} leaves.

Theorem 4. *There are* 2^{m-1} *permutations of* $\{1, \ldots, m\}$ *with* $[t]_\approx = \{t\}$.

Although exponential, this is a fast decreasing fraction of the size of the solution space $m!$ as can be seen as follows. Recall that without loss of generality m is even.

$$\begin{aligned} m! &= 2^{m-1}\left(\frac{m}{2}\right)! \prod_{i=1}^{m/2-1} (i + 1/2) \\ &> 2^{m-1}\left(\frac{m}{2}\right)!\left(\frac{m}{2} - 1\right)! \,. \end{aligned}$$

So, e.g., for $m = 48$ this means $2^{m-1}/m! \leq 10^{-46}$.