

# A Specialized Evolutionary Approach to the bi-objective Travelling Thief Problem

Maciej Laszczyk

Wrocław University of Science and Technology  
Faculty of Computer Science and Management  
ul. Ignacego Łukasiewicza 5, 50-371 Wrocław, Poland  
maciej.laszczyk@pwr.edu.pl

Paweł B. Myszkowski

Wrocław University of Science and Technology  
Faculty of Computer Science and Management  
ul. Ignacego Łukasiewicza 5, 50-371 Wrocław, Poland  
pawel.myszkowski@pwr.edu.pl

**Abstract**—In the recent years, it has been shown that real world-problems are often comprised of two, interdependent subproblems. Often, solving them independently does not lead to the solution to the entire problem. In this article, a Travelling Thief Problem is considered, which combines a Travelling Salesman Problem with a Knapsack Problem. A Non-Dominated Sorting Genetic Algorithm II (NSGA-II) is investigated, along with its recent modification - a Non-Dominated Tournament Genetic Algorithm (NTGA). Each method is investigated in two configurations. One, with generic representation, and genetic operators. The other, specialized to the given problem, to show how the specialization of genetic operators leads to better results. The impact of the modifications introduced by NTGA is verified. A set of Quality Measures is used to verify the convergence, and diversity of the resulting PF approximations, and efficiency of the method. A set of experiments is carried out. It is shown that both methods work almost the same when generic representation is used. However, NTGA outperforms classical NSGA-II in the specialized results.

## I. INTRODUCTION

**N**P-HARD optimization problems occur in many real-world scenarios. Be it a lot-sizing problem in economics [1], a transportation problem [2], or a scheduling problem [3]. These problems are ubiquitous and very practical, which makes their solving an important task. In practice, a problem often has multiple objectives. In scheduling problems, both time and cost of the schedule can be considered. In finance, it is desired to maximize the profits, but also to minimize the potential risks. Hence, multi-objective approaches aim to find a set of equally-good solutions, called a Pareto Front (PF).

Recently, authors of paper [4] have pointed out, that real-world problems comprise of multiple subproblems. They contain many dependencies and interwovenness. For that reason, it is not sufficient to find the solution to each of the subproblems. Objectives are interconnected in a way, that the improvement to one objective can lead to the worse value of another objective. Hence, combinations of such solutions do not guarantee the optimal solution to the entire problem. Authors of [5] proposed a Travelling Thief Problem (TTP), which has the features of a real-world problem. In this article, it is used in carried out experiments. It comprises of two constrained problems - a Travelling Salesman Problem (TSP) and a Knapsack Problem (KP). They are interconnected in a way, that makes solving them separately ineffective.

Due to its interdependence and multi-objective nature, evolutionary approaches show great potential in solving the TTP [6], [7], [8]. In [4] it has been shown that, in case of TTP, classical Non-Dominated Sorting Genetic Algorithm (NSGA-II) [9] with specialized operators outperforms other methods. Authors of [10] have introduced a Non-Dominated Tournament Genetic Algorithm. It is based on a NSGA-II, but contains multiple modifications. The authors carry out the research on a scheduling problem to show that these modifications lead to increased effectiveness of the method. This article attempts to verify the effectiveness of modifications introduced by NTGA with the combination of operators specialized for TTP.

A set of quality measures (QMs) proposed in [11] is used to evaluate the multi-objective results. Convergence and diversity of the resulting PF approximations are measured, along with the efficiency of the method.

The rest of the article is structured as follows. Section II presents existing work related to the subject. The TTP is described in section III. Section IV presents both used methods, as well as generic, and specialized representation, and operators. Results of all experiments along with the visualizations are presented in section V. Moreover, a theoretical analysis of the results is given. Lastly, section VI presents the conclusion and outlines the future work.

## II. RELATED WORK

The TTP was first introduced in [5]. The authors pointed out the shortcomings of benchmark problems used in the literature. The important features of a real-world problems were identified, namely existence of the subproblems and their interwovenness. Eventually, a single- and multi-objective versions of TTP were proposed.

Authors of [12] introduced a benchmark dataset for TTP. It contains 9720 instances. Each instance contains a TSP and KP elements. Additionally, the items are assigned to the cities to create the TTP instance. There are three different weight-value correlations present in the dataset. Moreover, instances contain up to 10 items per city.

Many researchers tackled the single-objective version of TTP. Authors of [13] have proposed three exact algorithms based on dynamic programming, branch and bound, and

constraint programming. Moreover, they compared them to the state-of-the-art solvers. In [8] swarm intelligence was used. Additionally, the authors investigated a TTP-specific local search algorithm. A Genetic Algorithm was used in [14]. Authors solve the overall problem instead of solving the subproblems separately. Moreover, the initial population is generated using a TSP specific heuristic. Authors of [18] used a hyperheuristic approach to select the best combination of known heuristics to solve the problem.

A multi-objective approach to TTP is considered less common in literature. However, authors of [15] used a combination of evolutionary computation and dynamic programming for the bi-objective TTP. Additionally, novel indicators were proposed, and the approach was compared to state-of-the-art methods. In [4], an NSGA-II with specialized representation and genetic operators was investigated. Various crossover and mutations method were investigated and the best configuration was identified. The results were compared to Greedy-based approaches. It was shown that NSGA-II outperforms other investigated methods.

NTGA was first proposed in [10]. The authors researched it on a bi-objective scheduling problem. The results were compared to classical NSGA-II and a decomposition based approach.

### III. PROBLEM

TTP is a constrained, combinatorial, NP-hard optimization problem. It comprises of two interwoven subproblems, namely TSP and KP. In the TSP part of the problem, there is a set of cities. Each city must be visited exactly once. In each of the cities there is a set of items, where each item has a weight and a value. Those items represent the KP part of the problem. While travelling a decision must be made which items to pick (if any).

TTP is a bi-objective problem. On one hand, the the goal is to find the quickest route between the cities. On the other, the total value of picked items must be maximized. However, each picked item decreases the speed of travel based on its weight. Hence, an improvement of the profit leads to an increase of the travelling time. TTP can be formally defined by equations 1 and 2.

$$\min f_{\tau}(\pi, z) \quad (1)$$

$$\max f_P(z) \quad (2)$$

where  $\pi$  is the permutation vector of all visited cities, and  $z$  is the picking plan.

The interaction between the subproblems is defined by equation 3. The total travelling time calculated as the sum of travelling times between each pair of consecutive cities plus the travelling time back to the first city. Each of those travelling times is influenced by all the picked items up to the given city.

$$f_{\tau}(\pi, z) = \sum_{i=1}^{n-1} \frac{d_{\pi_i, \pi_{i+1}}}{v(w(\pi_i))} + \frac{d_{\pi_n, \pi_1}}{v(w(\pi_n))} \quad (3)$$

$d_{\pi_i, \pi_{i+1}}$  is the distance between two consecutive cities from the permutation vector.  $n$  is the number of all cities.  $v(w(\pi_i))$  is the velocity in city  $\pi_i$ , considering the current weight  $w$  of picked items, and is defined by equation 4.

$$v(w) = v_{max} - \frac{W_c}{W}(v_{max} - v_{min}) \quad (4)$$

$W_c$  is the current weight, which is the sum of weights of all currently picked items.  $W$  is the capacity of the knapsack.  $v_{max}$  and  $v_{min}$  define the maximum and minimum allowed speed respectively.

The second objective, total profit, is the sum of values of all picked items. It is described by equation 5.

$$f_P(z) = \sum_{j=1}^m z_j z_j^{profit} \quad (5)$$

$m$  is the number of all items,  $z_j$  defines whether  $j$ 'th item has been picked and is equal to either 0 or 1.  $z_j^{profit}$  is the profit of  $j$ 'th item.

Additional constraints must be satisfied for the TTP solution to be *feasible*. The route must contain all cities, and each city must be visited exactly once. The sum of weights of all picked items cannot be greater than the capacity of the knapsack.

### IV. APPROACH

All approaches researched in the article are described in this section. First, definitions of important terms are given, namely dominance relation and a Pareto Front (PF). Next subsection describes used representation of an individual and is divided into two parts. First, a generic representation is described, and then one specialized for TTP. Similarly, the description of genetic operators starts with the operators used with generic representation. Later, the description of operators specialized for TTP is provided. Eventually, descriptions of NSGA-II and NTGA close out the section.

#### A. Definitions of Terms

This subsection contains a description of important terms. They are relevant for all used approaches.

1) *Dominance Relation*: One of the challenges of multi-objective optimization is the comparison of two solutions. Each solution is described by more than one objective, hence a numerical comparison is not sufficient. A Dominance Relation is defined for that purpose. A solution dominates another, if it has the value of at least one objective better and value of no objective worse, that that solution.

2) *Pareto Front*: A true PF contains all globally non-dominated solutions. However, in practice, a true PF is often not known. Hence, in this article, PF refers to the approximation found by the method, which contains all found non-dominated solutions.

### B. Representation

An individual used in evolutionary algorithms consists of a vector of numbers, a genotype. It represents the solution to the given problem. In this article, two representations are used. A generic one, presented in [10], and specialized for TTP, presented in [4].

1) *Generic*: The genome comprises two parts. The first one represents the solution to TSP. It assigns each city a priority. Then, the travelling plan is built by ordering the cities by that priority. The second part of the genome defines the solution to KP. It contains the number of genes equal to the number of items. Each item is assigned either 0 or 1, which defines whether the item should be picked or not.

For example a genome [1, 3, 3, 2, 0, 0, 1, 1] for a problem with 4 cities and 4 items means that the first city is visited first, then the last one, then second, and then third. Additionally, only the last 2 items are picked.

2) *Specialized*: Specialized representation defines the travelling plan with a permutation vector of all cities. The second part of genome is the same as in the case of generic representation.

For example a genome [1, 2, 4, 3, 1, 0, 0, 1] for a problem with 4 cities and 4 items means that the the first city is visited first, then the second one, then the last one, and finally the third one. The first and the last items are picked.

### C. Genetic Operators

This section describes the genetic operators used in the methods, for both representations.

Initial population generation is common for all methods and does not depend on the representation. A random initialization is used.

Selection operator is similarly independent of the representation, but is different for each method and is described in the appropriate method section (IV-D1 and IV-E1).

1) *Generic*: In case of generic representation, a standard single crossover and mutation operator is used for both parts of the genome. In the case of NSGA-II a single-point crossover is used. First, a random cut-point is selected within the genome. The first child is created by copying the genes on the left of that point from the first parent and copying the rest from the second parent. The second child is created similarly, by first copying the genes from the second parent and then from the first one. NTGA utilizes a single-point crossover.

Both methods use the same random mutation operator. First, a random gene is selected. Next, its value is randomly changed to a different, valid domain value.

2) *Specialized*: In the case of specialized representation, to include a problem domain knowledge a different crossover and mutation operator is used for each part of the genome. The Edge Operator (introduced in [16]) is used as the crossover for the part responsible for TSP. It aims to introduce as few as possible additional paths. It does so, by reusing existing edges when generating the children. First, a list is generated, which contains the neighbour cities from both parents for each city. Then, the first city from the first parent is copied to the

child genome and it is removed from the neighbour list. Then, iteratively, neighbours of that city with the fewest neighbours are copied over and removed from the list consecutively. If there are no more neighbours for a given city, a random city is selected. The second child is created similarly, by starting the process with the first city from the second parent. For the KP part of the genome, a uniform crossover is used. Children are created by copying each gene from a random parent with equal probability.

A swap mutation is utilized for the TSP part of the genome. Two random cities are selected and their position in the genome is swapped. For the KP part, a *bitflip* mutation is used. A random item is selected and the value of its gene is flipped.

In generic representation, there may occur a situation that makes an individual, a not *feasible* one. For example, some cities may have the same priority (cities are visited in defined order). In specialized representation crossover/mutation assure the feasibility of TSP-part of genome. However, another situation may exist in both representations when items picked by individual exceed knapsack capacity – items with min profit/weight ratio are removed from the solution.

### D. Non-Dominated Sorting Genetic Algorithm II

This section contains the description of a Non-Dominated Sorting Genetic Algorithm II (NSGA-II). It starts with the description of a pseudocode. Then, the selection and crowding distance, which are unique for this method are described.

NSGA-II is an evolutionary method. It processes a population of individuals in an iterative manner. Each individual represents a single solution to the given problem. The algorithm runs for the predefined number of generations, where a generation is a process of creating an offspring population from the current population. Each generation utilizes genetic operators to select parents and generate children individuals. Eventually, all non-dominated individuals found during the computation constitute a PF approximation. NSGA-II is described in pseudocode 1.

A *PopulationSize* parameter is stored in the first line. Then, in the second line, an initial population of that size is generated. In the third line, the entire population is evaluated. Each individual gets assigned the values of all objectives. Line 4 uses a non-dominated sorting to sort the population based on the rank and crowding distance, which are described in sections IV-D2 and IV-D3 respectively. In line 5, the loop begins, which iterates over all generations. Then, line 6 begins the loop to effectively double the size of the population. First, 2 parents are selected in line 7, using a selection described in IV-D1. In line 8, the crossover is used to create 2 children individuals. They are then mutated in line 9. Finally, the children are evaluated in line 10, and added to the current population in line 11. After the size of population has been doubled, it is again sorted in line 13. The population is truncated to its original size in line 14. Only the better half

**Algorithm 1** Pseudocode of NSGA-II [9]

---

```

1:  $N \leftarrow PopulationSize$ 
2:  $P_{current} \leftarrow generateInitialPopulation(N)$ 
    $evaluate(P_{current})$ 
3:  $nonDominatedSorting(P_{current})$ 
   for  $i \leftarrow 0$  to  $generationLimit$  do
4:   while  $|P_{current}| < 2N$  do
      $parents \leftarrow select(P_{current})$ 
5:      $children \leftarrow crossover(parents)$ 
      $children \leftarrow mutate(children)$ 
6:      $evaluate(children)$ 
      $P_{current} \leftarrow P_{current} \cup children$ 
7:   end while
    $nonDominatedSorting(P_{current})$ 
8:    $truncate(P_{current}, N)$ 
9: end for
10: return  $P_{current}$ 

```

---

of the population remains. After all the generations have been processed, the last population is returned in line 16.

1) *Selection*: Selection in NSGA-II starts with taking 2 random individuals from the population. Then, those 2 individuals are compared based on the rank and crowding distance, which are described in sections IV-D2 and IV-D3 respectively. The individual with the lower rank is selected. If both have the same rank, individual with the larger crowding distance is selected. Selection returns only one parent, so during the algorithm it is performed twice, to obtain 2 parents.

2) *Rank*: During the NSGA-II each individual is assigned a rank, based on its quality. The lower rank means the individual is better. It is computed in an iterative manner. First, all non-dominated individuals are assigned the rank equal to 1. Then, all individuals that remain-dominated, while not considering the individuals with the rank already set, have the rank set to 2. The process is repeated, until all individuals are assigned a rank. Intuitively, the process divides the population into multiple PF approximations. The rank describes to which of those approximations the individual belongs.

3) *Crowding Distance*: Crowding distance is calculated for each individual. First, the largest possible box is drawn around the individual, that contains only that individual from the population. The crowding distance is the volume of that box. The larger values mean that the individual lies in the poorly explored part of the space. At the same time, lower values mean that there are many individuals around given individual.

### E. Non-Dominated Tournament Genetic Algorithm

This section contains the description of a Non-Dominated Tournament Genetic Algorithm (NTGA). First a pseudocode is given. Then its selection and clone elimination methods are described.

NTGA is based on a classical NSGA-II method. It introduces 4 modifications that aim to improve the effectiveness of the method. First, it separates parent and child populations. Then, it utilizes a selection method with stronger selective

pressure. Finally, it introduces a clone elimination method and archive usage. NTGA is presented in pseudocode 2.

**Algorithm 2** Pseudocode of NTGA [10]

---

```

1:  $N \leftarrow PopulationSize$ 
2:  $archive \leftarrow \emptyset$ 
    $P_{current} \leftarrow generateInitialPopulation(N)$ 
3:  $evaluate(P_{current})$ 
    $updateArchive(P_{current})$ 
4: for  $i \leftarrow 0$  to  $generationLimit$  do
    $P_{next} \leftarrow \emptyset$ 
5:   while  $|P_{next}| < |P_{current}|$  do
      $parents \leftarrow select_{tour}(P_{current})$ 
6:      $children \leftarrow crossover(parents)$ 
      $children \leftarrow mutate(children)$ 
7:     while  $P_{next}$  contains  $children$  do
        $children \leftarrow mutate(children)$ 
8:     end while
      $evaluate(children)$ 
9:      $P_{next} \leftarrow P_{next} \cup children$ 
      $updateArchive(children)$ 
10:   end while
    $P_{current} \leftarrow P_{next}$ 
11: end for
12: return  $archive$ 

```

---

First line stores the *PopulationSize* parameter. An empty archive is initialized in line 2. It is designed to store all found non-dominated individuals. In line 3, an initial population of given size is created. It is then evaluated in line 4. In line 5, the archive is updated with all currently non-dominated individuals. The loop, in line 6, iterate over a predefined number of generations. In line 7, an empty population is initialized, which is going to store the next population. The loop, in line 8, runs until the size of next population is equal to the size of current population. In line 9, parents are selected with the selection method described in IV-E1. Then, the children are created with the crossover in line 10. They are mutated in line 11. The clone elimination method is described between lines 12 and 13. If the next population already contains generated children, they are mutated. After that, the children are evaluated in line 15. Finally, they are added to the next population in line 16. In line 17 the archive is updated. The children are added to it, if they are non-dominated. Then, all individuals, that the children dominate, are removed from the archive. When the next population has been fully generated, it replaces the current population. At the end, the archive of non-dominated individuals is returned. The archive contains the PF approximation.

1) *Selection*: NTGA uses a tournament selection. First, given number of individuals is randomly drawn from the population. Then, they are compared according to their rank (described in IV-D2). The individual with the lowest rank is selected. If there are multiple individuals that match, the first one is selected. It is worth noting that the crowding distance is not considered during the comparison.

2) *Clone Elimination*: Clone elimination aims to increase the diversity of the population. The clones are defined as individuals with the identical genome. Before adding a child individual to the population, a check is performed, to verify whether an identical individual already exists. If so, the child is mutated until it is no longer a clone. Only then, it is added to the next population.

## V. EXPERIMENTS AND RESULTS

The experiments carried out in this article aim to verify the effectiveness of NTGA on TTP. Moreover, it is verified whether the modifications of NTGA are effective in the context of specialized operators. To answer those question the experiments on 4 configurations are carried out. NSGA-II and NTGA are researched with both generic, and specialized representation, and operators. A set of selected QMs is used to verify the convergence and diversity of the resulting PF approximations.

First, selected data instances and quality measures are described. Then, the experimental procedure is presented and selected parameter values are provided. A full set of experiments is carried out on all 4 configurations and the results are presented. Finally, the last subsection contains the theoretical analysis.

### A. Data Instances

A benchmark dataset, first presented in [12], is used in this article. In literature, an *eil51* instances are often used [17] [18], and so they have been selected for the research. 12 instances have been selected with 51 cities and the number of items between 50 and 500. 3 types of correlation between item weight and profit can be identified within the set. A strong correlation, where increased profit also means increased value. No correlation, but all the items have similar weights. The last group has no correlation between the items.

### B. Quality Measures

A set of QMs presented in [11] is used to verify the effectiveness of the methods. Convergence, diversity of the PF approximation, and the efficiency of the method is verified. A Perfect Point and a Nadir Point are used as a reference in 2 of the measures.

1) *Perfect Point*: A Perfect Point contains the best values of all objectives. It does not have to an achievable solution. The value of travelling time is calculated as the length of minimum spanning tree of the tour. A brute force search algorithm is used to calculate the value of profit.

2) *Nadir Point*: A Nadir Point contains the worst values of all objectives from among the non-dominated solutions. It often has to be approximated. Additionally, to make the comparison fair even worse values can be selected [19]. The value of travelling time is calculated by taking the value of travelling time from a Perfect Point and doubling it. It is an upper bound of the TSP. The profit is set to 0 and represents a solution where no items are picked.

3) *Euclidean Distance*: Euclidean Distance (*ED*) is a measure of convergence. It shows how close the PF approximation is to the true PF. Since the true PF is not known, *ED* utilizes the Perfect Point. Value of *ED* is obtained by calculating the average distance between every point on the PF approximation and the Perfect Point. It can be formally defined by equation 6.

$$ED(PF) = \frac{\sum_{i=1}^{|PF|} d_i}{|PF|} \quad (6)$$

*PF* is the Pareto Front,  $d_i$  is the distance from the  $i$ 'th point to the Perfect Point.

4) *Hypervolume*: Hypervolume (*HV*) is a measure of diversity. It is a volume of a hypercube defined by the Nadir Point and the PF approximation. It measures the spread, but is also influenced by the convergence and uniformity of the PF approximation. *HV* can be formally defined by equation 7.

$$HV(PF) = \Lambda\left(\bigcup_{s \in PF} \{s' | s \prec s' \prec s^{nadir}\}\right) \quad (7)$$

*PF* is an approximation of PF.  $s$  is the point of approximated PF.  $s^{nadir}$  is a *Nadir Point*.  $\Lambda$  is a Lebesgue measure, which is the generalization of a volume.  $\prec$  is a domination relation.

5) *Pareto Front Size*: Pareto Front Size (*PFS*) measures the diversity in terms of the cardinality of the PF approximation. It is defined as the number of points on the PF approximation.

6) *Ratio of Non-Dominated Individuals*: Ratio of Non-Dominated Individuals (*RNI*) measure the efficiency of the method. It is defined as the number of points on the PF approximation divided by the number of all visited points.

7) *Spacing*: Spacing (*S*) measures the uniformity of the PF approximation. It ensures that the solutions are evenly distributed and identifies the clustering effect. To calculate it, first, the distances between all consecutive points on the PF approximation are calculated. The standard deviation of those distances is the *S* measure. It can be defined with equation 8.

$$S(PF) = \sqrt{\frac{1}{|PF|} \sum_{i=1}^{|PF|} (d_i - \bar{d})^2} \quad (8)$$

*PF* is the approximation of the PF.  $d_i$  is the distance from the  $i$ -th point to the next consecutive point.

The intuition of QMs is: *ED* should be minimized and measures the closeness to the true PF. *HV* should be maximized and it is influenced by both spread of the PF approximation and its distance to the true PF. *PFS* is simply the cardinality of the approximation, while *RNI* measures efficiency, by calculating the ratio of points on the approximation to all explored points. Spacing (*S*) should be minimized and it measures how closely the approximation resembles the uniform distribution.

### C. Experimental Procedure

First, the parameters of all methods have been tuned separately. Then, both NSGA-II and NTGA have been ran with both representations, on every instance. Due to the stochastic nature of evolutionary algorithms, each experiment has been repeated 50 times and results have been averaged. Next, a set of selected QMs has been calculated for each PF approximation. Statistical significance has been verified. Eventually, visualizations for selected instances are presented and theoretical analysis is described.

### D. Parameters

The first step in parameter tuning was to define a set of configurations of different parameter values. Taguchi Orthogonal Arrays have been used for that purpose. Then, each configuration has been ran 10 times and the QMs have been calculated for the resulting PF approximations. Next, a Multi-Objective Grey Relational Grade is calculated and a Taguchi Method is used to identify the impact of the parameters on the results [20]. Finally, the best parameter configuration is selected. This process has been repeated for all 4 configurations. The experiments show that a 1000 generations should be sufficient, however the limit has been set to 2000 to make sure the results converge.

Table I contains the selected configurations of parameter values for each researched method.

### E. Experiments

Tables II and III contain the results of experiments for NSGA-II and NTGA respectively. Both tables show the results on generic representation. Tables IV and V contain the results for NSGA-II and NTGA with specialized representation.

1) *Results*: Comparison of the results for the generic representation shows no significant difference between the results of QMs. The largest difference can be observed for *HV* for the benefit of NSGA-II, but it is within a single standard deviation.

Specialized operators improve the results significantly. The largest difference can again be observed for *HV*. NSGA-II with specialized operators have achieved better values of *HV* for all researched instances. Specialization has improved ED for 6 out of 12 instances. Overall, smaller instances show more improvement in terms of ED. Interestingly, *PFS* has been decreased almost 6 times. However, large values for generic representation might suggest that the solutions are far from local optima. There is no statistical difference in values of *RNI*. Values of *S* have also deteriorated. The largest difference can be seen for instances *eil51\_n250\_uncorr\_01* and *eil51\_n500\_uncorr\_01*. The difference can be justified by much larger spread of the approximation, which can be confirmed by the larger values of *HV*.

NTGA with specialized operators improves the results even further. On average, values of ED have been improved by almost 40%. The largest difference can be observed for larger instances. Values for *eil51\_n50\_bounded-strongly-corr\_01* and *eil51\_n50\_uncorr-similar-weights\_01* are worse than in case of NSGA-II. Better values of *HV* have been achieved for all

12 instances, which suggest much better diversity of the PF approximations. Similarly, larger *PFS* has been achieved for all 12 instances. For *eil51\_n250\_uncorr-similar-weights\_01* the value has been almost tripled. NTGA has also proved to be more efficient. It can be observed by larger values of *RNI*. The only instance that has not been improved is *eil51\_n50\_uncorr-similar-weights\_01*. Achieved values of *S* measure are also lower, however they are still almost 3 times larger than the values achieved for configurations with generic representation. Results compared to specialized versions of NSGA-II and NTGA presented in this section have been statistically confirmed by Wilcoxon signed-rank ( $W_{0.05} = 78 > W_c = 13$ ) for all QM's. All difference are statistically significant.

2) *Visualizations*: This section contains visualization for two selected instances. Figure 1 presents instance *eil51\_n50\_bounded-strongly-corr\_01*. It presents the case, where specialized NTGA has achieved the worse result than specialized NSGA-II in terms of ED measure. ED depends on a Perfect Point, which lies closest to the middle of PF. NSGA-II has achieved a large spread, but its approximation has very few points on the edges. Hence, average distance to the Perfect Point is relatively low. PF approximation generated by NTGA is more evenly distributed, and so many points lie far from the Perfect Point. Hence, ED deteriorates.

Figure 2 presents the second selected instance. For *eil51\_n500\_uncorr-similar-weights\_01* specialized NTGA improved the results the most in comparison to other configurations. Interestingly, specialized NSGA-II has generated the solutions with better profit than specialized NTGA.

Moreover, visualization of all achieved PF approximations for instance *eil51\_n250\_bounded-strongly-corr\_01* is presented in Figure 3. A modified version of empirical attainment function (EAF) [21] is used to get the "averaged" Pareto Front approximations. For clarity, only specialized configurations are shown. It can be seen, that NTGA achieves better results on average. Additionally, the deviations in the results are also smaller. However, NSGA-II has generated points with very high profit, that have not been dominated by any of the runs of NTGA.

### F. Summary

Table VI contains the summary of all obtained results for all configurations. NTGA does not improve the results in case of generic representation. Values of all measures are very similar for both NSGA-II and NTGA.

Specialization has improved the convergence and diversity of the PF approximation. It can be observed by the improved values of *ED* and *HV*. Specialization has decreased the value of *PFS*. However, in case of generic representation, achieved solutions are far from optimal, so their larger number is less significant. Specialized representation also led to increased distances between the points of the PF approximation. It might have been caused by the larger achieved spread.

In case of specialized representation, NTGA has improved the results significantly, even in comparison to specialized

TABLE I  
SELECTED PARAMETER CONFIGURATIONS

	representation	populationSize	generationLimit	$P_{m_{tsp}}$	$P_{m_{kp}}$	$P_{x_{tsp}}$	$P_{x_{kp}}$	tournamentSize
NSGA-II	generic	200	2000	0.005	0.005	0.9	0.9	2
	specialized	100	2000	0.1	0.05	0.6	0.8	2
NTGA	generic	50	2000	0.005	0.005	0.9	0.9	6
	specialized	50	2000	0.1	0.05	0.6	0.8	6

TABLE II  
VALUES OF SELECTED QMS FOR NSGA-II WITHOUT SPECIALIZATION

Instance	ED		HV		PFS		RNI		S	
	avg	std	avg	std	avg	std	avg	std	avg	std
eil51_n50_bounded-strongly-corr_01	0.4255	0.0486	0.7865	0.0297	65.4	8.3	0.0002	0.0000	0.0075	0.0029
eil51_n50_uncorr-similar-weights_01	0.3452	0.0338	0.6275	0.0122	3.3	1.6	0.0000	0.0000	0.0148	0.0109
eil51_n50_uncorr_01	0.3093	0.0319	0.8121	0.0064	34.0	8.2	0.0001	0.0000	0.0100	0.0025
eil51_n150_bounded-strongly-corr_01	0.3124	0.0156	0.7644	0.0198	169.0	46.6	0.0004	0.0001	0.0029	0.0011
eil51_n150_uncorr-similar-weights_01	0.3106	0.0390	0.6896	0.0586	47.4	27.6	0.0001	0.0001	0.0051	0.0025
eil51_n150_uncorr_01	0.2351	0.0130	0.7975	0.0131	90.6	25.2	0.0002	0.0001	0.0036	0.0011
eil51_n250_bounded-strongly-corr_01	0.2924	0.0111	0.7297	0.0173	206.1	57.1	0.0005	0.0001	0.0019	0.0009
eil51_n250_uncorr-similar-weights_01	0.3129	0.0175	0.7162	0.0152	97.3	29.5	0.0002	0.0001	0.0035	0.0012
eil51_n250_uncorr_01	0.2291	0.0108	0.8029	0.0160	144.5	33.1	0.0004	0.0001	0.0017	0.0005
eil51_n500_bounded-strongly-corr_01	0.2647	0.0119	0.7157	0.0186	253.7	101.1	0.0006	0.0003	0.0009	0.0002
eil51_n500_uncorr-similar-weights_01	0.3208	0.0190	0.6967	0.0139	162.1	57.2	0.0004	0.0001	0.0022	0.0010
eil51_n500_uncorr_01	0.2588	0.0086	0.7578	0.0076	189.9	56.5	0.0005	0.0001	0.0011	0.0006
Average	0.3014	0.0217	0.7414	0.0190	121.9	37.7	0.0003	0.0001	0.0046	0.0021

TABLE III  
VALUES OF SELECTED QMS FOR NTGA WITHOUT SPECIALIZATION

Instance	ED		HV		PFS		RNI		S	
	avg	std	avg	std	avg	std	avg	std	avg	std
eil51_n50_bounded-strongly-corr_01	0.4216	0.0451	0.7591	0.0330	55.4	13.5	0.0001	0.0000	0.0084	0.0052
eil51_n50_uncorr-similar-weights_01	0.3307	0.0247	0.6439	0.0155	5.3	2.2	0.0000	0.0000	0.0140	0.0103
eil51_n50_uncorr_01	0.2951	0.0290	0.8175	0.0103	31.0	8.4	0.0001	0.0000	0.0109	0.0042
eil51_n150_bounded-strongly-corr_01	0.3119	0.0147	0.7441	0.0218	144.3	52.0	0.0004	0.0001	0.0038	0.0016
eil51_n150_uncorr-similar-weights_01	0.3123	0.0334	0.6932	0.0383	49.1	25.2	0.0001	0.0001	0.0049	0.0019
eil51_n150_uncorr_01	0.2376	0.0118	0.7893	0.0123	86.6	28.2	0.0002	0.0001	0.0029	0.0008
eil51_n250_bounded-strongly-corr_01	0.2963	0.0140	0.7118	0.0110	202.4	79.0	0.0005	0.0002	0.0022	0.0012
eil51_n250_uncorr-similar-weights_01	0.3057	0.0195	0.7052	0.0166	93.0	39.8	0.0002	0.0001	0.0030	0.0010
eil51_n250_uncorr_01	0.2343	0.0148	0.7885	0.0143	116.3	40.5	0.0003	0.0001	0.0021	0.0021
eil51_n500_bounded-strongly-corr_01	0.2726	0.0097	0.6965	0.0171	263.0	105.6	0.0007	0.0003	0.0011	0.0011
eil51_n500_uncorr-similar-weights_01	0.3397	0.0276	0.6669	0.0335	159.9	73.5	0.0004	0.0002	0.0021	0.0010
eil51_n500_uncorr_01	0.2608	0.0185	0.7481	0.0165	214.8	51.9	0.0005	0.0001	0.0009	0.0004
Average	0.3016	0.0219	0.7303	0.0200	118.4	43.3	0.0003	0.0001	0.0047	0.0026

NSGA-II. Both ED and HV values have been improved. Additionally, efficiency of the algorithm has been improved, which can be observed by the increased value of RNI. Value of S measure has been improved in comparison to specialized NSGA-II. However, it remains higher than in case of generic representation.

### G. Theoretical Analysis

In NTGA, parent individuals do not have to compete with children individuals. There is no possibility that an individual will survive for multiple generations. Hence, more unique points are explored by the method. In combination with increased selective pressure it also leads to increased convergence. Interestingly, introduction of clone prevention has not improved the diversity. Larger values of HV are caused by the larger distance from the Nadir Point and not by the larger spread of the approximation. More significant improvement can be observed for larger instances.

Modifications of NTGA lead to no significant improvement in case of generic representation. Non-specialized operators have a low probability of improving the result. In consequence, increased selective pressure has much less significance.

The crowding distance has been removed from NTGA. Instead, each new individual has to be compared with the existing individuals to verify whether it is a clone. In most cases the comparison is done only once. However, if the individual is a clone, it is mutated and the check is performed again. In an edge case the comparison must be done multiple times, which might negatively affect the performance.

## VI. CONCLUSIONS AND FUTURE WORK

In this paper NTGA has been investigated in the context of TTP. A bi-objective problem, which comprises of two subproblems. The subproblems are interconnected, which makes solving them independently ineffective. NTGA has been compared to classical NSGA-II. All experiments have been carried out

TABLE IV  
VALUES OF SELECTED QMs FOR NSGA-II WITH SPECIALIZED REPRESENTATION

Instance	ED		HV		PFS		RNI		S	
	avg	std	avg	std	avg	std	avg	std	avg	std
eil51_n50_bounded-strongly-corr_01	0.3239	0.0484	0.8492	0.0092	30.2	6.3	0.0002	0.0000	0.0301	0.0140
eil51_n50_uncorr-similar-weights_01	0.4049	0.0592	0.7106	0.0067	9.8	2.7	0.0000	0.0000	0.0639	0.0286
eil51_n50_uncorr_01	0.2094	0.0172	0.8444	0.0091	11.6	3.5	0.0001	0.0000	0.0188	0.0072
eil51_n150_bounded-strongly-corr_01	0.2315	0.0240	0.8094	0.0131	33.5	8.4	0.0002	0.0000	0.0233	0.0299
eil51_n150_uncorr-similar-weights_01	0.2283	0.0289	0.7877	0.0127	16.0	7.3	0.0001	0.0000	0.0312	0.0469
eil51_n150_uncorr_01	0.2016	0.0506	0.8202	0.0101	16.5	4.5	0.0001	0.0000	0.0379	0.0642
eil51_n250_bounded-strongly-corr_01	0.2349	0.0492	0.8060	0.0169	31.7	10.2	0.0002	0.0001	0.0466	0.0459
eil51_n250_uncorr-similar-weights_01	0.2890	0.1075	0.7956	0.0138	16.9	8.7	0.0001	0.0000	0.0829	0.0801
eil51_n250_uncorr_01	0.2756	0.0942	0.8269	0.0099	18.4	5.4	0.0001	0.0000	0.1072	0.0785
eil51_n500_bounded-strongly-corr_01	0.3336	0.0805	0.7776	0.0157	33.4	11.1	0.0002	0.0001	0.0800	0.0397
eil51_n500_uncorr-similar-weights_01	0.4108	0.0836	0.8049	0.0094	26.9	10.2	0.0001	0.0001	0.0894	0.0253
eil51_n500_uncorr_01	0.4596	0.1021	0.8136	0.0090	25.0	8.1	0.0001	0.0000	0.1372	0.0395
Average	0.3003	0.0621	0.8038	0.0113	22.5	7.2	0.0001	0.0000	0.0624	0.0417

TABLE V  
VALUES OF SELECTED QMs FOR NTGA WITH SPECIALIZED REPRESENTATION

Instance	ED		HV		PFS		RNI		S	
	avg	std	avg	std	avg	std	avg	std	avg	std
eil51_n50_bounded-strongly-corr_01	0.3881	0.0424	0.8752	0.0098	51.3	10.6	0.0005	0.0001	0.0211	0.0078
eil51_n50_uncorr-similar-weights_01	0.4234	0.0738	0.7365	0.0039	12.9	4.2	0.0001	0.0000	0.0533	0.0145
eil51_n50_uncorr_01	0.2205	0.0221	0.8786	0.0052	19.2	4.3	0.0002	0.0000	0.0167	0.0063
eil51_n150_bounded-strongly-corr_01	0.2107	0.0190	0.8513	0.0112	67.1	18.1	0.0007	0.0002	0.0058	0.0025
eil51_n150_uncorr-similar-weights_01	0.2199	0.0373	0.8286	0.0105	37.5	17.8	0.0004	0.0002	0.0129	0.0060
eil51_n150_uncorr_01	0.1544	0.0062	0.8553	0.0065	31.6	8.8	0.0003	0.0001	0.0045	0.0026
eil51_n250_bounded-strongly-corr_01	0.1777	0.0151	0.8447	0.0119	67.8	16.4	0.0007	0.0002	0.0053	0.0063
eil51_n250_uncorr-similar-weights_01	0.1821	0.0183	0.8471	0.0096	53.9	18.3	0.0005	0.0002	0.0073	0.0028
eil51_n250_uncorr_01	0.1478	0.0068	0.8639	0.0063	33.9	8.8	0.0003	0.0001	0.0031	0.0025
eil51_n500_bounded-strongly-corr_01	0.1645	0.0103	0.8265	0.0102	70.6	17.8	0.0007	0.0002	0.0041	0.0070
eil51_n500_uncorr-similar-weights_01	0.1659	0.0131	0.8459	0.0098	64.9	22.0	0.0006	0.0002	0.0051	0.0047
eil51_n500_uncorr_01	0.1553	0.0114	0.8442	0.0081	36.5	9.9	0.0004	0.0001	0.0084	0.0294
Average	0.2175	0.0230	0.8415	0.0086	45.6	13.1	0.0005	0.0001	0.0123	0.0077

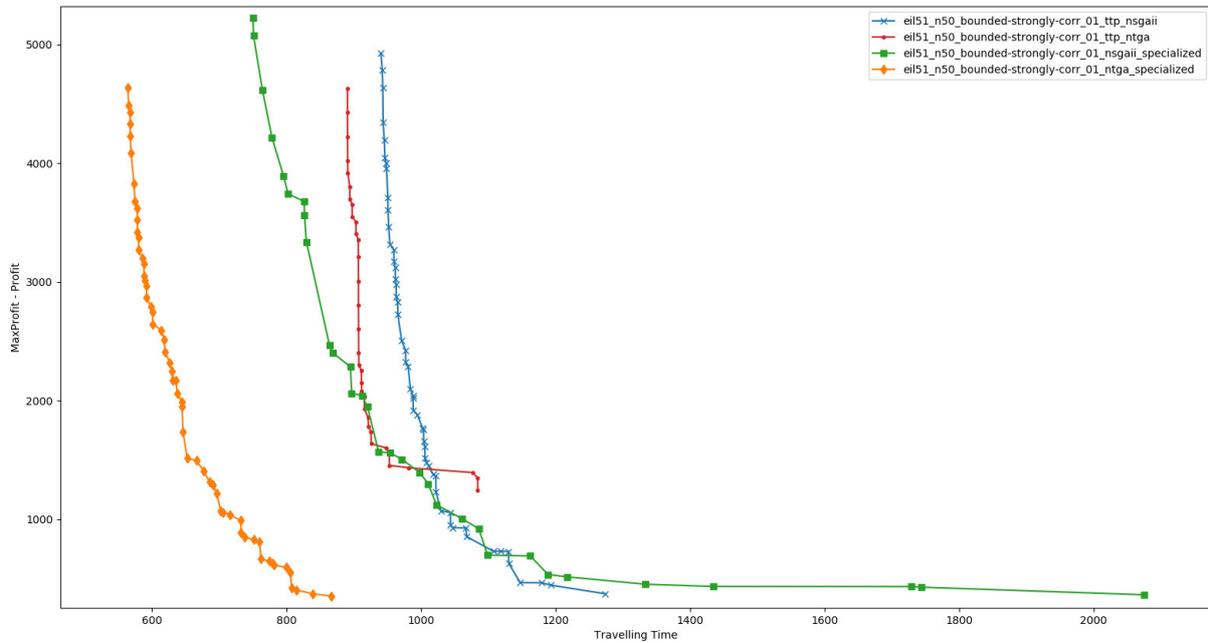


Fig. 1. Comparison of selected approx. Pareto Fronts for data instance eil51\_n50\_bounded-strongly-corr\_01

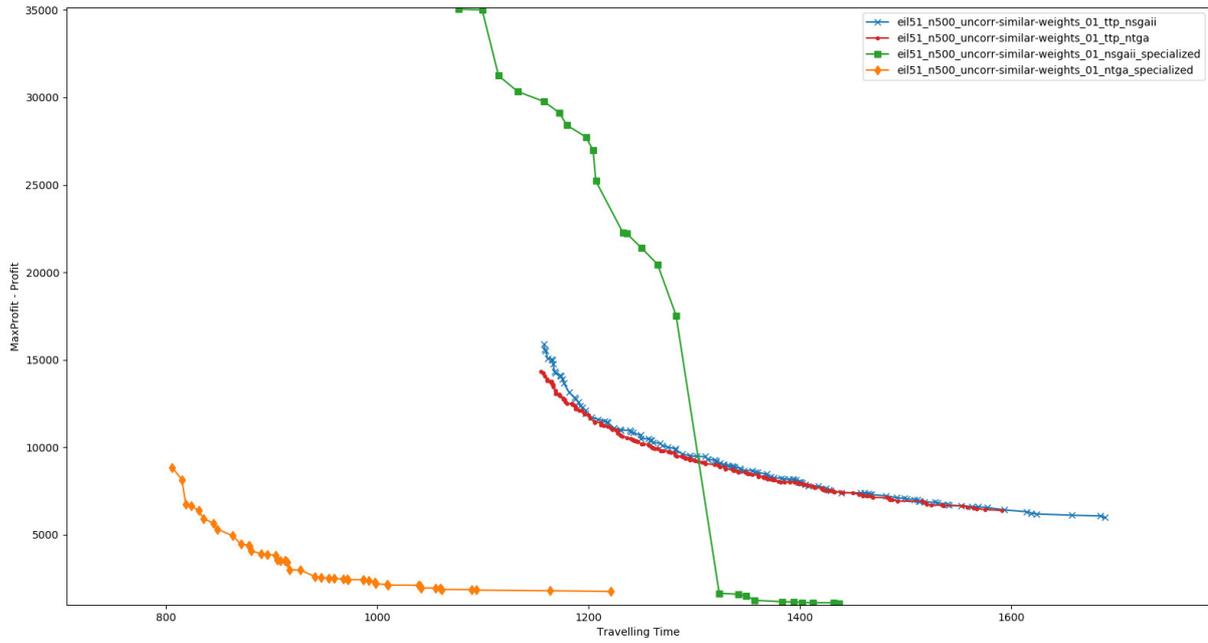


Fig. 2. Comparison of selected approx. Pareto Fronts for data instance eil51\_n500\_uncorr-similar-weights\_01

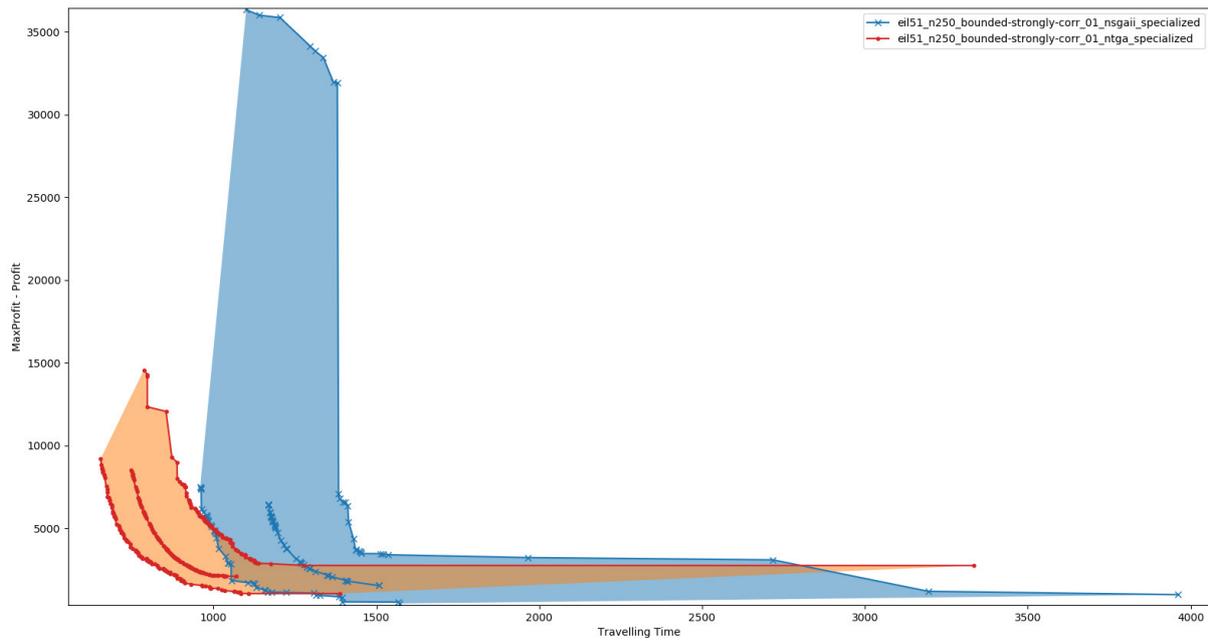


Fig. 3. Comparison of EAF average approx. Pareto Fronts for data instance eil51\_n250\_bounded-strongly-corr\_01

TABLE VI  
SUMMARY OF ALL RESULTS

		ED		HV		PFS		RNI		S	
		avg	std	avg	std	avg	std	avg	std	avg	std
NSGA-II	generic	0.3014	0.0217	0.7414	0.0190	121.9	37.7	0.0003	0.0001	0.0046	0.0021
	specialized	0.2991	0.0622	0.8029	0.0124	22.3	6.9	0.0002	0.0001	0.0659	0.0431
NTGA	generic	0.3016	0.0219	0.7303	0.0200	118.4	43.3	0.0003	0.0001	0.0047	0.0026
	specialized	0.2158	0.0208	0.8419	0.0084	45.4	12.2	0.0005	0.0001	0.0120	0.0079

in configurations with generic, and specialized representation. It has been shown, that for larger instances specialized NTGA achieves better results than specialized NSGA-II.

Increased selective pressure of NTGA led to improved results. However, more research could be done regarding selection, that would also promote diversity of the PF approximation. An introduction of heuristics, that would further improve the solutions for the subproblems might be worth investigating. Additionally, a hyperheuristic that would combine the benefits of multiple evolutionary methods could prove beneficial to the results. Moreover, many-objective problems are fairly uncommon. An interesting avenue of future work would be to use a benchmark problem with the real-world characteristics, with a larger number of objectives.

#### REFERENCES

- [1] Akbalik, Ayse, et al. "NP-hard and polynomial cases for the single-item lot sizing problem with batch ordering under capacity reservation contract." *European Journal of Operational Research* 257.2 (2017): 483-493.
- [2] Sanei, Masoud, et al. "Step fixed-charge solid transportation problem: a Lagrangian relaxation heuristic approach." *Computational and Applied Mathematics* 36.3 (2017): 1217-1237.
- [3] Mnich, Matthias, and Rene van Bevern. "Parameterized complexity of machine scheduling: 15 open problems." *Computers & Operations Research* (2018).
- [4] Blank, Julian, Kalyanmoy Deb, and Sanaz Mostaghim. "Solving the Bi-objective Traveling Thief Problem with Multi-objective Evolutionary Algorithms." *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, Cham, 2017.
- [5] Bonyadi, Mohammad Reza, Zbigniew Michalewicz, and Luigi Barone. "The travelling thief problem: The first step in the transition from theoretical problems to realistic problems." *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013.
- [6] Bonyadi, Mohammad Reza, et al. "Evolutionary computation for multi-component problems: opportunities and future directions." *Optimization in Industry*. Springer, Cham, 2019. 13-30.
- [7] Martins, Marcella SR, et al. "HSEDA: a heuristic selection approach based on estimation of distribution algorithm for the travelling thief problem." *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2017.
- [8] Wagner, Markus. "Stealing items more efficiently with ants: a swarm intelligence approach to the travelling thief problem." *International Conference on Swarm Intelligence*. Springer, Cham, 2016.
- [9] Deb, Kalyanmoy, et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II." *IEEE transactions on evolutionary computation* 6.2 (2002): 182-197.
- [10] Laszczyk, Maciej, and Paweł B. Myszkowski. "Improved selection in evolutionary multi-objective optimization of Multi-Skill Resource-Constrained project scheduling problem." *Information Sciences* 481 (2019): 412-431.
- [11] Laszczyk, Maciej, and Paweł B. Myszkowski. "Survey of quality measures for multi-objective optimization. Construction of complementary set of multi-objective quality measures." *Swarm and Evolutionary Computation* (2019).
- [12] Polyakovskiy, Sergey, et al. "A comprehensive benchmark set and heuristics for the traveling thief problem." *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. ACM, 2014.
- [13] Wu, Junhua, et al. "Exact approaches for the travelling thief problem." *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer, Cham, 2017.
- [14] Vieira, Daniel KS, et al. "A genetic algorithm for multi-component optimization problems: the case of the travelling thief problem." *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, Cham, 2017.
- [15] Wu, Junhua, et al. "Evolutionary computation plus dynamic programming for the bi-objective travelling thief problem." *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2018.
- [16] Whitley, L. Darrell, Timothy Starkweather, and D'Ann Fuquay. "Scheduling problems and traveling salesmen: The genetic edge recombination operator." *ICGA*. Vol. 89. 1989.
- [17] Marti, Luis, Eduardo Segredo, and Emma Hart. "Impact of selection methods on the diversity of many-objective Pareto set approximations." *Procedia Computer Science* 112 (2017): 844-853.
- [18] Yafrani, Mohamed, et al. "A hyperheuristic approach based on low-level heuristics for the travelling thief problem." *Genetic Programming and Evolvable Machines* 19.1-2 (2018): 121-150.
- [19] Cao, Yongtao, Byran J. Smucker, and Timothy J. Robinson. "On using the hypervolume indicator to compare Pareto fronts: Applications to multi-criteria optimal experimental design." *J. of Stat. Planning and Inference* 160 (2015): 60-74.
- [20] Durairaj, M., D. Sudharsun, and N. Swamynathan. "Analysis of process parameters in wire EDM with stainless steel using single objective Taguchi method and multi objective grey relational grade." *Procedia Engineering* 64 (2013): 868-877.
- [21] Lopez-Ibanez M., Paquete L., and Stutzle T. "Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization", *Experimental Methods for the Analysis of Optimization Algorithms* (2010): 209-222.