Evolutionary Computation plus Dynamic Programming for the Bi-Objective Travelling Thief Problem

Junhua Wu
Optimisation and Logistics
School of Computer Science
University of Adelaide

Sergey Polyakovskiy
School of Information Technology
Deakin University

Markus Wagner
Optimisation and Logistics
School of Computer Science
University of Adelaide

Frank Neumann
Optimisation and Logistics
School of Computer Science
University of Adelaide

ABSTRACT

This research proposes a novel indicator-based hybrid evolutionary approach that combines approximate and exact algorithms. We apply it to a new bi-criteria formulation of the travelling thief problem, which is known to the Evolutionary Computation community as a benchmark multi-component optimisation problem that interconnects two classical $\mathcal{NP}$-hard problems: the travelling salesman problem and the 0-1 knapsack problem. Our approach employs the exact dynamic programming algorithm for the underlying packing while travelling problem as a subroutine within a bi-objective evolutionary algorithm. This design takes advantage of the data extracted from Pareto fronts generated by the dynamic program to achieve better solutions. Furthermore, we develop a number of novel indicators and selection mechanisms to strengthen synergy of the two algorithmic components of our approach. The results of computational experiments show that the approach is capable to outperform the state-of-the-art results for the single-objective case of the problem.

CCS CONCEPTS

• **Mathematics of computing → Combinatorial optimization**; • **Theory of computation → Evolutionary algorithms**; Dynamic programming; • **Applied computing** → *Multi-criterion optimization and decision-making*;

KEYWORDS

Bi-objective optimisation; Genetic algorithms; Dynamic programming; Travelling thief problem; Multi-component problem; Hybrid approach

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07...$15.00
https://doi.org/10.1145/3205455.3205488

ACM Reference Format:
Junhua Wu, Sergey Polyakovskiy, Markus Wagner, and Frank Neumann. 2018. Evolutionary Computation plus Dynamic Programming for the Bi-Objective Travelling Thief Problem. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205488

1 INTRODUCTION

The travelling thief problem (TTP) [4] is a bi-component problem, where two well-known $\mathcal{NP}$-hard combinatorial optimisation problems, namely the travelling salesperson problem (TSP) and the 0-1 knapsack problem (KP), are interrelated. Hence, tackling each component individually is unlikely to lead to a global optimal solution. It is an artificial benchmark problem modelling features of complex real-world applications emerging in the areas of planning, scheduling and routing. For example, Stolk et al. [24] exemplify a delivery problem that consists of a routing part for the vehicle(s) and a packing part of the goods onto the vehicle(s).

Thus far, many approaches have been proposed for the TTP [5, 6, 10–12, 14–18, 22, 25, 27, 31, 32, 34]. However, to the best of our knowledge, all of them focus on utilising the existing heuristic approaches (such as local search, simulated annealing, tabu search, genetic algorithms, memetic algorithm, swarm intelligence, etc.), incorporating either well-studied operators of the TSP and KP or slight variations of such operators. The heuristic approaches or operators that take advantage of the existing exact algorithms of the TTP [20, 30] are yet lacking. In addition, very few investigations have considered multi-objective formulations of the TTP except those studied by Blank et al. [3] and Yafrani et al. [33].

In this paper, we consider a bi-objective version of the TTP, where the goal is to minimise the weight and maximise the overall benefit of a solution. We present a hybrid approach for the bi-objective TTP that uses the dynamic programming approach for the underlying packing while travelling (PWT) problem as a subroutine. The evolutionary component of our approach constructs a tour π for the TTP. This tour is then fed into the dynamic programming algorithm to compute a trade-off front for the bi-objective problem. Here, the tour π is kept fixed and the resulting packing solutions are Pareto optimal owing to the capability of the dynamic programming. A key aspect of the algorithm is to take advantage of the different fronts belonging to different tours for the TTP component, as presumably the global Pareto front might contain segments from different fronts. Meanwhile, when the evolutionary approach evolves the tours and the current general Pareto front consists of different tours (together with the packing plans), a challenge is to select tours for mutations and crossovers that lead to promising new tours. Such tours can result in new Pareto optimal solutions for the overall bi-objective TTP problem when running the dynamic programming on them. In short, the selection mechanism shall encourage the synergy of the two sub-approaches. We introduce a novel indicator-based evolutionary algorithm (IBEA [35]) that contains a series of customised indicators and parent selections to achieve this goal. Our results show that this approach solves the problem well, and its by-product, which is the total reward of the single objective TTP, beats the state-of-the-art approach in most cases.

The remainder of the paper first states the bi-objective version of the TTP mathematically in Section 2. Then, Section 3 covers the prerequisites required for our approach, which is later introduced in Section 4. Section 5 provides the description of the computational setup and the analysis of computational experiments. Finally, Section 6 draws conclusions.

2 THE TRAVELLING THIEF PROBLEM

The standard single-objective formulation of the TTP [22] involves n cities, m items, and a thief who must make a tour visiting each of the cities exactly once. The cities form a set of nodes $V = \{1, \ldots, n\}$ in a complete graph $G = (V, E)$, where $E \subseteq V^2$ is a set of edges representing all possible connections between the cities. Every edge $e_{ij} \in E$ is assigned a known distance d_{ij}. Every node $i \in V$ but the first one relates to a unique set of items $M_i = \{1, \ldots, m_i\}$, $\sum_{i=2}^{n} m_i = m$, stored in the corresponding city. Each item $k \in M_i$ positioned in node i is associated with an integer profit p_{ik} and an integer weight w_{ik}. The thief starts and ends the tour in the first node and can collect any of the items located in the intermediate nodes $2, \ldots, n$. Items may only be selected as long as their total weight does not exceed the knapsack's capacity C. Furthermore, the thief pays a rent rate R for each time unit of travelling. Selection of an item contributes its profit to a total reward, but produces a transportation cost relative to its weight. As the weight of each added item slows down the thief, the transportation cost increases. This cost is therefore deducted from the reward. When the knapsack is empty, the thief can achieve a maximal velocity υ_{max}. When it is full, the thief can only move with a minimal velocity $\upsilon_{min} > 0$. The actual velocity υ_i when moving along the edge e_{ij} depends on the total weight of items chosen in the cities preceding i. The problem asks to determine a combination of a tour and a subset of items that maximizes the overall profit, which is defined as the value of the selected items minus the overall transportation cost.

Let an integer-valued vector $\pi \in V^n$, $\pi = (\pi_1, ..., \pi_n)$, represent a tour such that $\pi_i = j$ iff j is the ith visited node of the tour. Clearly, $\pi_i \neq \pi_j$ for any $i, j \in V$, $i \neq j$. Next, let a binary decision vector $\rho \in \{0,1\}^m$, $\rho = (\rho_{21}, ..., \rho_{nm_n})$, encode a packing plan of the problem such that $\rho_{ik} = 1$ iff item k in node i is chosen, and 0 otherwise. Then $W_{\pi_i} = \sum_{j=1}^{i} \sum_{k=1}^{m_j} w_{jk}\rho_{jk}$ is a total weight of items sequentially selected in the nodes from π_1 to π_i, and $\upsilon_{\pi_i} = \upsilon_{max} - \nu W_{\pi_i}$, $\nu = (\upsilon_{max} - \upsilon_{min})/C$, is the real velocity of the thief leaving the ith node. In summary, the objective function of the TTP has the following form:

$$f(\pi, \rho) = \sum_{i=1}^{n} \sum_{k=1}^{m_i} p_{ik}\rho_{ik} - R\left(\sum_{i=1}^{n-1} \frac{d_{\pi_i \pi_{i+1}}}{\upsilon_{\pi_i}} + \frac{d_{\pi_n \pi_1}}{\upsilon_{\pi_n}}\right) \tag{1}$$

Here, we extend the standard formulation of the TTP by introduction of an additional objective function. The new version, named as BO-TTP for short, becomes a bi-objective optimisation problem, where the total accumulated weight

$$\varphi(\rho) = \sum_{i=1}^{n} \sum_{k=1}^{m_i} w_{ik}\rho_{ik} \tag{2}$$

yields the second criterion. Such an extension appears natural regarding the TTP as one may either need to maximise the reward for a given weight of collected items, or determine the least weight subject to bounds imposed on the reward. Note that even if π is fixed, (1) is a non-monotone sub-modular function [23] that implies possible deterioration of the reward as the number of selected items, and therefore their total weight, increases. We formulate the BO-TTP as follows:

$$(\pi, \rho) = \begin{cases} \arg\max f(\pi, \rho) \\ \arg\min \varphi(\rho) \end{cases} \quad s.t.\ \varphi(\rho) \leq C$$

As a bi-objective optimisation problem, BO-TTP asks for a set of Pareto-optimal solutions, where each feasible solution cannot be improved in a second objective without degrading quality of the first one, and vice versa. In other words, the goal is to find a set of all non-dominated feasible solutions $X \subseteq \Pi \times P$ such that for any solution $(\pi, \rho) \in X$ there is no solution $(\pi', \rho') \in X$ such that either $(f(\pi, \rho) < f(\pi', \rho')) \wedge (\varphi(\rho) \leq \varphi(\rho'))$ or $(f(\pi, \rho) \geq f(\pi', \rho')) \wedge (\varphi(\rho) > \varphi(\rho'))$ holds, where Π is a set of feasible tours and P is a set of feasible packing plans.

3 PREREQUISITES

The PWT is a special case of the TTP, which maximises the total reward for a specific tour π [23]. Thus, an optimal solution of the PWT defines a subset of items producing the maximal gain. This yields a non-linear knapsack problem, which can be efficiently solved via the dynamic programming (DP) approach proposed by Neumann et al. [20]. Most importantly, the DP yields not just a single optimal packing plan, but a set of plans $\bar{P}_\pi \subseteq P$, where (π, ρ) and (π, ρ') do not dominate each other for any $\rho, \rho' \in \bar{P}_\pi$. We name the corresponding objective vectors of $\bar{P}_\pi$ as a *DP front*. In Section 4, we design our hybrid algorithm that takes advantage of the features of a DP front.

For self-sufficiency of the paper, in Section 3.1, we first briefly explain the DP and how we adopt it to obtain a DP front. Section 3.2 then discusses several algorithms to obtain tours that are later utilised by the DP to create multiple DP fronts and to initialise the population for our hybrid evolutionary approach.

3.1 Dynamic Programming for the PWT

The DP for the PWT bases on a scheme traditional to the classical 0-1 knapsack problem. It processes items in the lexicographic order as they appear along a given tour π; that is, item $l \in \pi_i$ strictly precedes item $k \in \pi_j$, to be written as $l \preceq k$, if either $\pi_i < \pi_j$ or $(\pi_i = \pi_j) \wedge (l \leq k)$ holds. Its table B is an $m \times C$ matrix, where entry β_{kw} represents the maximal reward that can be achieved by examining all combinations of items l with $l \leq k$ leading to the weight equal to w. The base case of the DP with respect to the first item k, according to the precedence order, positioned in node π_i is as follows:

$$\beta_{kw} = \begin{cases} -\frac{R}{\upsilon_{max}} \left(\sum_{j=1}^{n-1} d_{\pi_j \pi_{j+1}} + d_{\pi_n \pi_1} \right), & \text{if } w = 0 \\ p_{\pi_i k} - R \left(\sum_{j=1}^{n-1} \frac{d_{\pi_j \pi_{j+1}}}{\upsilon_{\pi_j}} + \frac{d_{\pi_n \pi_1}}{\upsilon_{\pi_n}} \right), & \text{if } w = w_{\pi_i k} \\ -\infty, & \text{if } w \notin \{0, w_{\pi_i k}\} \end{cases}$$

Here, the first case relates to the empty packing when the thief collects no items at all while travelling along π, and the second computes the reward when only item k is chosen. Where a combination yielding w does not exist, $\beta_{kw} = -\infty$. For the general case, let item l be the predecessor of item k with regard to the precedence order. And let $\beta(k\cdot)$ denote the column containing all the entries β_{kw} for $w \in [0, C]$. Then based on $\beta(l\cdot)$ one can obtain $\beta(k\cdot)$ computing each entry β_{kw}, assuming that item k is in node π_i, as

$$max \begin{cases} \beta_{lw} \\ \beta_{l\,w-w_{\pi_i k}} + p_{\pi_i k} - R \sum_{j=i}^{n-1} \left(\frac{d_{\pi_j \pi_{j+1}}}{\upsilon_{max} - \nu w} - \frac{d_{\pi_j \pi_{j+1}}}{\upsilon_{max} - \nu(w - w_{\pi_i k})} \right) \\ \quad -R \left(\frac{d_{\pi_n \pi_1}}{\upsilon_{max} - \nu w} - \frac{d_{\pi_n \pi_1}}{\upsilon_{max} - \nu(w - w_{\pi_i k})} \right) \end{cases}$$

In order to reduce the search space, in each column the cells dominated by other cells are to be eliminated, i.e. if $\beta_{kw_1} > \beta_{kw_2}$ and $w_1 \leq w_2$, then $\beta_{kw_2} = -\infty$. An optimal solution derived by the DP corresponds to the maximal reward stored in the last column of B. That is, $\max_w \{\beta(s, w)\}$ is the value of an optimal solution, where s is the last item according to the precedence order.

The last column of B can be considered as leading to a complete set of non-dominated packing plans $\bar{P}_\pi \subseteq P_\pi \subseteq P$, where P_π is the set of all feasible packing plans for a given tour π.

Definition 3.1. Let τ_π be a corresponding objective vector for $\bar{P}_\pi$. Then τ_π represents the related Pareto front designated as a *DP front* for the given tour π.

In fact, the DP front τ_π for a tour π is a complete non-dominated set as it contains all non-dominated objective vectors with regard to $\bar{P}_\pi$. We take advantage of this completeness of a DP front to generate a variety of solutions in our bi-objective approach in Section 4.

3.2 Generation of Multiple DP Fronts

As a single DP front τ_π is produced for a single given tour π, i.e. $\pi \mapsto \tau_\pi$, we could generate multiple TSP tours to get a set of DP fronts. In practice, various algorithms are capable of producing superior tours for the TSP, and therefore many approaches to the TTP use this capability to succeed. High-performing TTP algorithms are commonly two-stage heuristic approaches, like those proposed by Polyakovskiy et al. [22], Faulkner et al. [12], and El Yafrani and Ahiod [10]. Specifically, their first step generates a near-optimal TSP tour and the second step completes solution by selection of a subset of items. Most of the approaches utilise the Chained Lin-Kernighan heuristic [2], because it is able to provide very tight upper bounds for TSP instances in short time. The knapsack component then is often handled via constructive heuristics or evolutionary approaches. However, the TTP is essentially structured in such a way that the importance of its both components is almost equal within the problem. Although near-optimal TSP solutions can give good solutions to the TTP, most of them are far away from being optimal [30]. This is the reason for our first experimental study, where we investigate the impact of several TSP algorithms on TTP solutions. Note that owing to the DP we are able to solve the knapsack part to optimality, which contributes to the validity of our findings.

We analysed five algorithms for the TSP: the Inver-over heuristic (INV) [26], the exact solver Concorde (CON) [1], the ant colony-based approach (ACO) [9], the Chained Lin-Kernighan heuristic (LKH) [2] and its latest implementation (LKH2) [13]. We ran each algorithm 10,000 times on every instance of the *eil76* series of the TTP benchmark suite [22]. We computed 100 (capped due to practical reasons) distinct tours by INV, 25 by CON, 24 by the both ACO and LKH, and 12 by LKH2. The lengths of the tours generated by INV are narrowly distributed around the average of 588.64 with the standard deviation being 2.55. By contrast, every other algorithm generates tours having the identical tour length of 585, which beats INV.

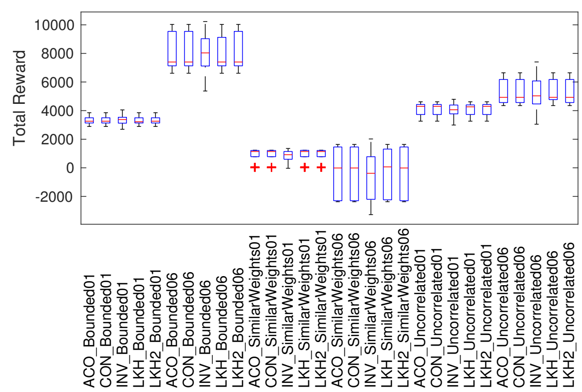


Figure 1: Exploring diversity of TSP tours on the *eil76_n75* series of the TTP instances.

We then applied the DP to every tour produced by each of the algorithms. Figure 1 depicts the resulted rewards on some sample TTP instances, where each box with whiskers reports the distribution of the rewards for a certain instance and the corresponding algorithm. The central mark of each box indicates the median of rewards, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers extend to the most extreme rewards without considering outliers, and outliers are plotted individually as plus signs. From the plot, we may observe that the tours generated by the CON, ACO, LKH and LKH2 have similar distributions of rewards. By contrast, the boxes of INV seem to be more extreme on the both sides. This means that the distribution of rewards via INV is more diverse and the best of the rewards outperform the others. In other words, though the Inver-over heuristic may lose against modern TSP approaches, it performs better in the role of generator of varied tours for the TTP. It may act as a seeding algorithm for a population in evolutionary algorithms. Note that this observation is in line with that by Wagner [27], who noticed that good TTP solutions can require slightly longer TSP tours.

In Figure 2, we visualise the collection of the DP fronts produced by the DP on the TTP instance *eil76_n75_uncorr_01* [22]. The corresponding tours are the 100 tours generated by the Inver-over heuristic. Actually, the plot depicts 200 fronts since the DP was applied to a tour and its reversed order.

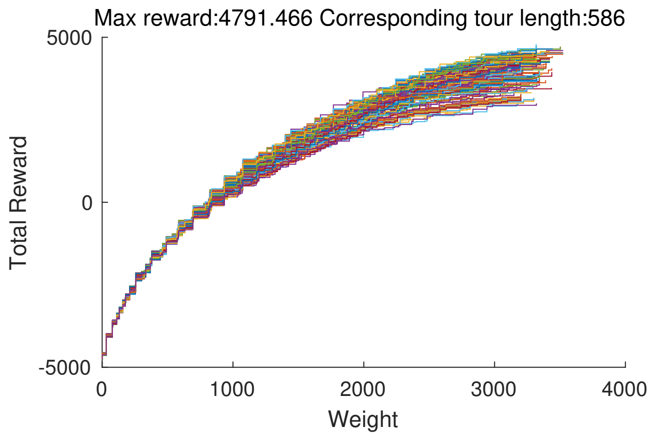


Figure 2: The visualisation of 200 DP fronts, generated according to 100 TSP tours produced by Inver-over for the TTP instance *eil76_n75_uncorr_01*.

Definition 3.2. Given q different DP fronts, let Φ denote a set of all possible unique solution points derived by $\tau_1, \dots, \tau_q$. Then ω is a Pareto front formed by the points of Φ and named as the *surface* of Φ.

The surface ω is formed by the union of all superior points resulted from different DP fronts in Φ. It is further used to guide evolution process in our approach.

4 A HYBRID EVOLUTIONARY APPROACH

Multi-objective optimisation algorithms guided by evolutionary mechanisms explore the decision space iteratively in order to determine a set of Pareto-optimal solutions. Indeed, many of them may act myopically as they sample the space searching for individual solutions without clear vision of the whole picture in terms of other solutions and their number. Therefore, achieving strong diversity in exploring the space plays an important role in evolutionary algorithm design. In this paper, we discuss one way to overcome potential issues related to diversity and propose a hybrid approach where evolutionary techniques and dynamic programming find synergy in their combination.

One of the challenges of multi-objective optimisation is to keep the variation of solutions large, that is to be a result of stronger diversity. Modern approaches normally incorporate additional processes to tackle this, such as the density estimation and/or crowdedness-comparison operator in SPEA2 [36] and NSGA-II [8]. In our approach, the DP is incorporated as a subroutine capable of producing at once a series of possible decisions with regard to a given tour. Thus, when a tour is specified, the DP guarantees that a corresponding Pareto front will be built without missing any of its points due to the completeness of the DP front, which thus also guarantees a good spread of solutions.

On the other hand, due to the typically observed non-dominance between individual DP fronts, the global Pareto-optimal front of the BO-TTP may be formed either by a single DP front or by a combination of segments from different top DP fronts. In Figure 2, we can observe that the DP fronts are all intertwined together, including the ones at the surface of the fronts collection. This indicates that the Pareto-optimal set of solutions is more likely to be the result of multiple TSP tours and their DP fronts. Our evolutionary mechanism takes advantage of this and keeps the superior DP fronts so that the population can be further improved. In order to achieve this as well as to overcome the drawback of existing multi-objective evolutionary optimisation algorithms focusing on individual solutions, we design our hybrid IBEA incorporating indicators and selection mechanisms aimed on orchestrating the improvement of the Pareto front, which are further guided by the DP fronts calculated for most promising TSP tours.

Our hybrid approach reduces the search space to some extent by decomposing the problem and thus transforming it. Evolutionary optimisation approaches traditionally depend on the choice of solution encoding (i.e. chromosome). Our approach treats a single TSP tour as an individual. Thus, a set of tours yields a population. Indeed, it operates on a reduced set of variables (implying shorter chromosomes), thus decreasing memory consumption and the number of internally needed sorting operations, comparisons and search procedures.

Algorithm 1 Hybrid IBEA Approach

Input: population size μ; limit on the number of generations α;
Initialisation:
set the iteration counter $c = 0$;
populate $\overline{\Pi}$ with μ new tours produced by the TSP solver;
while $(c \leq \alpha)$ **do**
  set $c = c + 1$;
  Indicator:
  run the DP for every tour $\pi \in \overline{\Pi}$ to compute its DP front τ_π;
  apply indicator function $\mathcal{I}(\tau_\pi)$ to calculate the indicator value for every individual tour $\pi \in \overline{\Pi}$;
  Survivor Selection:
  repeatedly remove the individual with the smallest indicator value from the population $\overline{\Pi}$ until the population size is μ (ties are broken randomly);
  Parent Selection:
  apply parent selection procedure to $\overline{\Pi}$ according to the indicator values to choose a set Λ of λ parent individuals;
  Mating:
  apply crossover and mutation operators to the parents of Λ to obtain a child population Λ';
  set the new population as $\overline{\Pi} = \overline{\Pi} \cup \Lambda'$;
end while

Algorithm 1 sketches the whole approach, which we adopted from the original IBEA introduced by Zitzler and Künzli [35]. It accepts μ as a control parameter for the size of the population

$\overline{\Pi} \subseteq \Pi$ and α as a limit on the number of iterations, which defines its termination criterion. In order to utilise the information within the DP fronts to guide the evolution of individual tours, we design new indicators to be computed based on the DP fronts instead of doing this directly on the individuals. Our specific selection mechanisms then filter the individuals according to the indicator values in order to find the tours with better DP fronts.

The rest of this section first introduces the indicator functions we apply to TSP solutions. Next, it details a parent selection mechanism to mate existing individuals from the population. It ends with a discussion of mutation and crossover operators guiding the search.

4.1 Design of Indicators

The designs of our indicators are based on the idea of measuring how much each DP front contributes to the surface ω produced by the set Φ (cf. the definition 3.2) corresponding to the population $\overline{\Pi}$. Given a DP front τ_π for a tour $\pi \in \overline{\Pi}$ and its corresponding set of solution points T_π, the value of the indicator $\mathcal{I}$ based on a measuring function $\mathcal{M}$ is to be calculated as follows:

$$\mathcal{I}(\tau_\pi) = 1 - \frac{\mathcal{M}(\Phi \setminus T_\pi)}{\mathcal{M}(\Phi)}. \tag{3}$$

This formula measures how much one loses (expressed as a value ranging from 0 to 1) should the points of the front τ_π be not included to the surface ω. We study two types of the measurement functions: Surface Contribution (SC) and Hypervolume (HV), hence two corresponding indicators: the Loss of Surface Contribution (LSC) and the Loss of Hypervolume (LHV).

Loss of Surface Contribution. Our first indicator is SC, which is a novel and direct measure. Given a DP front τ_π and its corresponding solution set T_π, $SC(T_\pi)$ counts the number of objective vectors that τ_π contributes to ω as defined by:

$$SC(T_\pi) = \frac{|\Phi \cap T_\pi|}{|\Phi|}. \tag{4}$$

Using SC (4) to replace the $\mathcal{M}$ function in (3), we obtain the formula for LSC as follows:

$$LSC(\tau_\pi) = 1 - SC(\Phi \setminus T_\pi). \tag{5}$$

Loss of Hypervolume. In multi-objective optimisation, the hypervolume indicator is a traditional indicator used to indicate the quality of a set of objective vectors [37]. In the bi-criteria case, when a front is given as a set of points in two-dimensional space, its value is computed as a sum of areas of rectangular regions.

Let $(0, C)$ be the reference point for our problem, which implies that only the range of non-negative objective values is taken into account. In addition, let $p = (u, v) \in \tau_\pi$ be a point in a DP front τ_π while $u > 0$ and $v < C$, and thus $p \in T_\pi$. Then $HV(T_\pi)$ calculates the hypervolume for τ_π as:

$$HV(T_\pi) = \sum_{p \in T_\pi} u_p \left(v_p - v_{p-1}\right) \tag{6}$$

Placing $HV(T_\pi)$ back to (3), we have the loss of hypervolume $LHV(\tau_\pi)$ computed as

$$LHV(\tau_\pi) = 1 - \frac{HV(\Phi \setminus T_\pi)}{HV(\Phi)}. \tag{7}$$

4.2 Parent Selection Mechanisms

With the individuals in the population $\overline{\Pi}$ being measured by the defined indicators, we can study strategies that shall efficiently select good individuals. There are five parent selection schemes that we take into consideration due to their popularity or previous theoretical findings. For the purpose of the later comparison, we introduce two simple selection rules as well as a traditional policy that should form a baseline. We aim to find a well-performing combination of an indicator and a selection rule to strengthen the synergy of the dynamic programming and evolutionary approach.

Rank-based Selection (RBS). In the rank-based selection policy, individuals are first ranked with respect to the value of an indicator. The selection policy is based then on a specific distribution law affecting the choice of a parent. Here, we study three schemes introduced by Osuna et al. [21], namely exponential (EXP), inverse quadratic (IQ) and Harmonic (HAR), and make them a part of our hybrid approach. Given a population of size μ, the probability of selecting the ith ranked individual according to EXP, IQ and HAR is, respectively,

$$\frac{2^{-i}}{\sum_{j=1}^{\mu} 2^{-j}}, \ \frac{i^{-2}}{\sum_{j=1}^{\mu} j^{-2}}, \ \frac{i^{-1}}{\sum_{j=1}^{\mu} j^{-1}}. \tag{8}$$

Fitness-Proportionate Selection (FPS). This rule estimates an individual $\pi \in \overline{\Pi}$ according to the indicator $\mathcal{I}(\tau_\pi)$ of its DP front τ_π. It has the following form:

$$FPS(\pi_i) = \frac{\mathcal{I}(\tau_i)}{\sum_{j=1}^{\mu} \mathcal{I}(\tau_j)}. \tag{9}$$

Tournament Selection (TS). This policy applies the tournament selection [19], but employs indicators discussed in Section 4.1 to rank individuals.

Arbitrary Selection (AS). Here, we consider two different rules: the best arbitrary selection (BST) and another one, which we call extreme (EXT). The former ranks individuals of a population with accordance to the value of an indicator and selects the best half of the population. The latter proceeds similarly selecting 25% of the best and 25% of the worst individuals.

Uniformly-at-random Selection (UAR). This traditional policy selects a parent from a population with probability $\frac{1}{\mu}$ uniformly at random.

4.3 Mutation and Crossover Operators

In our approach, we adopt a multi-point crossover operator that has already proved its efficiency for the TTP in [10]. As an (unoptimised) rule, we perform the crossover operation on a tour with 80% probability. It is always followed by the mutation procedure, which either applies the classical 2OPT mutation [7] or re-inserts a node to another location. Both the node and the location are selected uniformly at random. We name these two operators 2OPT and JUMP, respectively.

5 COMPUTATIONAL EXPERIMENTS

5.1 Computational Set Up

We examine the IBEA presented in Algorithm 1 by combining each of the two indicators with each of the eight parent selection rules.

This results in 16 different settings. For example, FPS on LHV means the combination of the FPS selection and the LHV indicator.

From the original set of the TTP instances, we use three different types, namely bounded-strongly-correlated (*Bounded*), uncorrelated (*Uncorrelated*) and uncorrelated-with-similar-weights (*SimilarWeights*), selected from the three series: *eil51, eil76, eil101* in the TTP benchmark [22]. We repeat our approach 30 times for each of the selected instances. Each time, the algorithm performs 20,000 generations on a population $\overline{\Pi}$ of size 50.

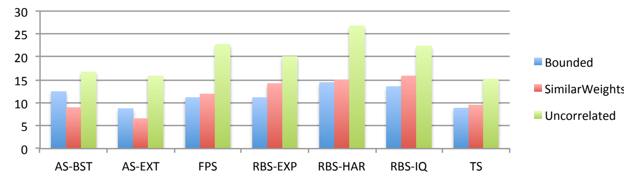
As the experiments are computationally expensive, we run them on the university's supercomputer consisting of 5568 Intel(R) Xeon(R) 2.30GHz CPU cores and 12TB of memory. Overall the experiments consumed around 170,000 CPU hours.

5.2 Results and Analysis

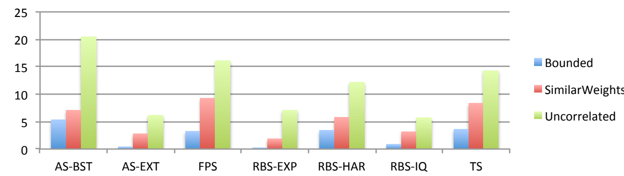
The comparison of our settings is based on the final populations and the hypervolumes derived for the surfaces of resulting non-dominated solutions. As the IBEA is able to solve the single-objective version of the TTP, we compare the total reward obtained to the result of the state-of-the-art approach MA2B [10] (see comparison in [28]). Because of varying mean values and unknown global optima solutions of the TTP benchmark instances, it is hard to analyse and perform comparison across them. However, such a comparison is desired as it may provide a more precise analysis of our algorithm. To resolve this issue, we proceed with a statistical comparison of the results. First, we use the UAR selection rule as the baseline that contributes to the two baseline settings: UAR on LHV and UAR on LSC. Second, we apply the *Welch's t-test* [29] to the two settings and the other variants.

The results of the t-test are probability values (p-values), each measures the likelihood of one selection rule to the corresponding baseline with regard to achieved performance. For example, we obtain the p-value equal 4.75×10^{-7} when compare the hypervolume of the FPS and the UAR on LHV. This means that the probability of the FPS performing similarly to the UAR on LHV (as expressed by having the same means) is less than 0.0000475%. In fact, the former performs much better than the latter on average. In order to improve the readability, we apply logarithm scaling to p-values in our plots. Thus, the measure of the FPS on LHV in our small example is 6.32 (i.e. $\log_{10}(4.75 \times 10^{-7})$). In fact, the larger the logarithmic p-value is, the better the selection rule performs comparing to the UAR.

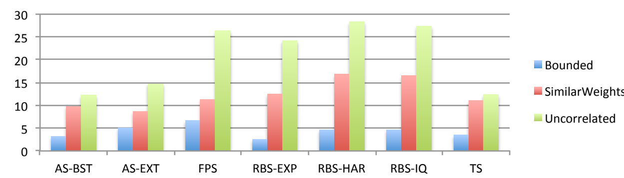
Figure 3 depicts the overall results of the Welch's t-test, in which we categorise our results into three types of bars according to three types of TTP instances: *Bounded*, *Uncorrelated* and *SimilarWeights*. Each bar of the plots represents the mean of the logarithmically scaled p-values of several instances in this category, for example *eil51_n50_bounded-strongly-corr_01.ttp*, *eil76_n75_bounded-strongly-corr_01.ttp* and *eil101_n100_bound-ed-strongly-corr_01.ttp*. This shows us different patterns between the selection rules running on the LHV and the LSC respectively. For example, the RBS schemes generally perform better on LHV than on LSC, among which the HAR is the best. Indeed, the HAR is the least aggressive scheme among these three, with a fat tail and relatively small probability for selecting the best few individuals [21]. It implies that the LHV benefits mainly result from the diversity of candidates. By contrast, the AS-BST are superior on LSC that likely implies that the LSC relies more on a few outstanding individuals for approximation, as the AS-BST only focuses on the best ones.



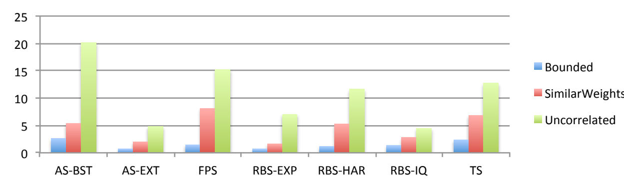
(a) Hypervolume of each selection on LHV



(b) Hypervolume of each selection on LSC



(c) Total Reward of each selection on LHV



(d) Total Reward of each selection on LSC

Figure 3: Investigating the performance of the selection rules via the Welch's t-test. P-values are logarithmically scaled. UAR selection serves as a base line.

In terms of different types of TTP instances, we may observe that the IBEA achieves the best results for the uncorrelated instances for all of the settings while being the worst on the strongly bounded ones for most of the settings. This to some extent supports the conjecture that strongly bounded TTP instances are the (relatively) hard ones and uncorrelated instances are the simplest to compute [22].

With regard to the choice of the parent selections, except for the RBS-HAR and the AS-BST, which show outperforming behaviour on LHV and LSC, respectively, we also recommend the FPS. This selection rule seems to be a reasonable choice as it performs consistently well on different settings.

Overall, we may observe from Figure 3 that the figures of the hypervolume generally confirm those for the total reward. This suggests that solving the bi-objective TTP also suggests good solutions for the single objective case. Table 1 presents the total rewards we achieve for various instances of the BO-TTP. They are compared to

MA2B				
		Mean	Max	SD
eil51_n50	Uncorrelated	2805.000	2855	27.814
	SimilarWeights	1416.348	1460	47.906
	Bounded	4057.652	4105	25.841
eil76_n75	Uncorrelated	5275.067	5423	78.138
	SimilarWeights	1398.867	1502	55.448
	Bounded	3849.067	4109	139.742
eil101_n100	Uncorrelated	3339.600	3789	388.360
	SimilarWeights	2215.500	2483	235.905
	Bounded	4949.000	5137	139.285

FPS LHV				
		Mean	Max	SD
eil51_n50	Uncorrelated	2828.728	2854.543	15.357
	SimilarWeights	1413.044	1459.953	17.780
	Bounded	4229.149	4230.997	10.118
eil76_n75	Uncorrelated	5445.624	5514.666	58.992
	SimilarWeights	1477.680	1513.404	24.494
	Bounded	4042.449	4108.760	38.805
eil101_n100	Uncorrelated	3620.844	3943.425	222.815
	SimilarWeights	2431.907	2482.462	52.265
	Bounded	5094.246	5233.513	65.267

FPS LSC				
		Mean	Max	SD
eil51_n50	Uncorrelated	2810.509	2832.496	18.076
	SimilarWeights	1426.135	1459.953	21.990
	Bounded	4231.299	4241.199	1.881
eil76_n75	Uncorrelated	5392.575	5514.666	73.029
	SimilarWeights	1474.803	1513.404	21.346
	Bounded	4054.815	4102.167	21.440
eil101_n100	Uncorrelated	3664.369	3846.172	124.994
	SimilarWeights	2436.374	2482.462	49.731
	Bounded	5067.070	5233.513	55.587

Table 1: Comparing the total reward obtained by the MA2B and the IBEA for the single-objective TTP. IBEA runs with the FPS selection rule and the LHV and LSC indicators, respectively. Results outperforming those of MA2B based on 30 restarts are highlighted.

the state-of-art MA2B [10] algorithm for the single objective TTP. Both MA2B and IBEA are given identical time limits. The results demonstrate that IBEA outperforms MA2B in the majority of the test cases.

6 CONCLUSION

In this paper, we investigated a new bi-objective travelling thief problem which optimises both the total reward and the total weight. We propose a hybrid indicator-based evolutionary algorithm that utilises the exact dynamic programming algorithm for the underlying PWT problem as a subroutine to evolve the individuals. This approach facilitates computation of a variety of feasible solutions without introducing additional diversification mechanisms. We furthermore design and study novel indicators and selection schemes that take advantage of the information extracted from the Pareto fronts generated by the exact approach and are used to evolve solutions towards the global Pareto optimality. The results of computational experiments show that our approach solves the problem efficiently that is further supported by the results outperforming those of the state-of-the-art approach for the single-objective TTP.

ACKNOWLEDGEMENTS

The authors were supported by Australian Research Council grants DP130104395 and DE160100850.

REFERENCES

[1] David Applegate, Ribert Bixby, Vasek Chvatal, and William Cook. 2005. Concorde tsp solver, 2006. *See: http://www.math.uwaterloo.ca/tsp/concorde.html* (2005).
[2] David Applegate, William J. Cook, and André Rohe. 2003. Chained Lin-Kernighan for Large Traveling Salesman Problems. *INFORMS Journal on Computing* 15, 1 (2003), 82–92.
[3] Julian Blank, Kalyanmoy Deb, and Sanaz Mostaghim. 2017. *Solving the Bi-objective Traveling Thief Problem with Multi-objective Evolutionary Algorithms.* Springer International Publishing, Cham, 46–60.
[4] Mohammad Reza Bonyadi, Zbigniew Michalewicz, and Luigi Barone. 2013. The travelling thief problem: The first step in the transition from theoretical problems to realistic problems. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013.* 1037–1044.
[5] Mohammad Reza Bonyadi, Zbigniew Michalewicz, Michal Roman Przybylek, and Adam Wierzbicki. 2014. Socially inspired algorithms for the travelling thief problem. In *Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014.* 421–428.
[6] Shelvin Chand and Markus Wagner. 2016. Fast Heuristics for the Multiple Traveling Thieves Problem. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016 (GECCO '16).* ACM, 293–300.
[7] Georges A Croes. 1958. A Method for Solving Traveling-Salesman Problems. *Operations Research* 6, 6 (1958), 791–812. https://doi.org/10.1287/opre.6.6.791
[8] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation* 6, 2 (2002), 182–197.
[9] Marco Dorigo and Thomas Stützle. 2004. *Ant colony optimization.* MIT Press.
[10] Mohamed El Yafrani and Belaïd Ahiod. 2016. Population-based vs. Single-solution Heuristics for the Travelling Thief Problem. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016.* ACM, 317–324.
[11] Mohamed El Yafrani, Marcella Martins, Markus Wagner, Belaïd Ahiod, Myriam Delgado, and Ricardo Lüders. 2017. A hyperheuristic approach based on low-level heuristics for the travelling thief problem. *Genetic Programming and Evolvable Machines* (15 7 2017).
[12] Hayden Faulkner, Sergey Polyakovskiy, Tom Schultz, and Markus Wagner. 2015. Approximate Approaches to the Traveling Thief Problem. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation.* ACM, New York, NY, USA, 385–392.
[13] Keld Helsgaun. 2000. An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research* 126, 1 (2000), 106–130.
[14] Nuno Lourenço, Francisco B. Pereira, and Ernesto Costa. 2016. *An Evolutionary Approach to the Full Optimization of the Traveling Thief Problem.* Springer International Publishing, Cham, 34–45.
[15] Marcella S. R. Martins, Mohamed El Yafrani, Myriam R. B. S. Delgado, Markus Wagner, Belaïd Ahiod, and Ricardo Lüders. 2017. HSEDA: A Heuristic Selection Approach Based on Estimation of Distribution Algorithm for the Travelling Thief Problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17).* ACM, New York, NY, USA, 361–368.
[16] Yi Mei, Xiaodong Li, Flora Salim, and Xin Yao. 2015. Heuristic evolution with Genetic Programming for Traveling Thief Problem. In *IEEE Congress on Evolutionary Computation, CEC 2015, Sendai, Japan, May 25-28, 2015.* 2753–2760.
[17] Yi Mei, Xiaodong Li, and Xin Yao. 2014. Improving Efficiency of Heuristics for the Large Scale Traveling Thief Problem. In *Simulated Evolution and Learning - 10th International Conference, SEAL 2014, Dunedin, New Zealand, December 15-18, 2014. Proceedings.* 631–643.
[18] Yi Mei, Xiaodong Li, and Xin Yao. 2016. On investigation of interdependence between sub-problems of the Travelling Thief Problem. *Soft Comput.* 20, 1 (2016), 157–172.
[19] Brad L. Miller and David E. Goldberg. 1995. Genetic Algorithms, Tournament Selection, and the Effects of Noise. *Complex Systems* 9, 3 (1995). http://www.complex-systems.com/abstracts/v09_i03_a02.html
[20] Frank Neumann, Sergey Polyakovskiy, Martin Skutella, Leen Stougie, and Junhua Wu. 2017. A Fully Polynomial Time Approximation Scheme for Packing While Traveling. *CoRR* abs/1702.05217 (2017). arXiv:1702.05217
[21] Edgar Covantes Osuna, Wanru Gao, Frank Neumann, and Dirk Sudholt. 2017. Speeding up evolutionary multi-objective optimisation through diversity-based parent selection. In *Proceedings of the Genetic and Evolutionary Computation*

Conference, Berlin, Germany, July 15-19, 2017. 553–560.

[22] Sergey Polyakovskiy, Mohammad Reza Bonyadi, Markus Wagner, Zbigniew Michalewicz, and Frank Neumann. 2014. A comprehensive benchmark set and heuristics for the traveling thief problem. In *Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014*. 477–484.

[23] Sergey Polyakovskiy and Frank Neumann. 2017. The Packing While Traveling Problem. *European Journal of Operational Research* 258, 2 (2017), 424–439.

[24] Jacob Stolk, Isaac Mann, Arvind Mohais, and Zbigniew Michalewicz. 2013. Combining vehicle routing and packing for optimal delivery schedules of water tanks. *OR Insight* 26, 3 (2013), 167–190. http://dx.doi.org/10.1057/ori.2013.1

[25] Anna Strzeżek, Ludwik Trammer, and Marcin Sydow. 2015. DiverGene: Experiments on controlling population diversity in genetic algorithm with a dispersion operator. In *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*. 155–162. https://doi.org/10.15439/2015F411

[26] Guo Tao and Zbigniew Michalewicz. 1998. Inver-over Operator for the TSP. In *Parallel Problem Solving from Nature - PPSN V, 5th International Conference, Amsterdam, The Netherlands, September 27-30, 1998, Proceedings*. 803–812.

[27] Markus Wagner. 2016. Stealing Items More Efficiently with Ants: A Swarm Intelligence Approach to the Travelling Thief Problem. In *Swarm Intelligence: 10th International Conference (ANTS)*. Springer, 273–281.

[28] Markus Wagner, Marius Lindauer, Mustafa Mısır, Samadhi Nallaperuma, and Frank Hutter. 2017. A case study of algorithm selection for the traveling thief problem. *Journal of Heuristics* (07 Apr 2017). https://doi.org/10.1007/s10732-017-9328-y

[29] Bernard L Welch. 1947. The Generalization of 'Student's' Problem when Several Different Population Variances are Involved. *Biometrika* 34, 1/2 (1947), 28–35. http://www.jstor.org/stable/2332510

[30] Junhua Wu, Markus Wagner, Sergey Polyakovskiy, and Frank Neumann. 2017. Exact Approaches for the Travelling Thief Problem. In *Simulated Evolution and Learning - 11th International Conference, SEAL 2017, Shenzhen, China, November 10-13, 2017, Proceedings*. 110–121.

[31] Mohamed El Yafrani and Belaïd Ahiod. 2017. A local search based approach for solving the Travelling Thief Problem: The pros and cons. *Appl. Soft Comput.* 52 (2017), 795–804.

[32] Mohamed El Yafrani and Belaïd Ahiod. 2018. Efficiently solving the Traveling Thief Problem using hill climbing and simulated annealing. *Information Sciences* 432 (2018), 231–244.

[33] Mohamed El Yafrani, Shelvin Chand, Aneta Neumann, Belaïd Ahiod, and Markus Wagner. 2017. Multi-objectiveness in the Single-objective Traveling Thief Problem. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, New York, NY, USA, 107–108.

[34] Mohamed El Yafrani, Marcella S. R. Martins, Markus Wagner, Belaïd Ahiod, Myriam Regattieri Delgado, and Ricardo Lüders. 2018. A hyperheuristic approach based on low-level heuristics for the travelling thief problem. *Genetic Programming and Evolvable Machines* 19, 1-2 (2018), 121–150.

[35] Eckart Zitzler and Simon Künzli. 2004. Indicator-Based Selection in Multiobjective Search. In *Parallel Problem Solving from Nature - PPSN VIII, 8th International Conference, Birmingham, UK, September 18-22, 2004, Proceedings*. 832–842.

[36] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. 2001. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-report* 103 (2001).

[37] Eckart Zitzler and Lothar Thiele. 1998. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Parallel Problem Solving from Nature - PPSN V, 5th International Conference, Amsterdam, The Netherlands, September 27-30, 1998, Proceedings*. 292–304.