Co-evolutionary Diversity Optimisation for the Traveling Thief Problem

Adel Nikfarjam[1(✉)], Aneta Neumann[1], Jakob Bossek[2], and Frank Neumann[1]

[1] Optimisation and Logistics, School of Computer Science, The University of Adelaide, Adelaide, Australia
adel.nikfarjam@adelaide.edu.au

[2] AI Methodology, Department of Computer Science, RWTH Aachen University, Aachen, Germany

Abstract. Recently different evolutionary computation approaches have been developed that generate sets of high quality diverse solutions for a given optimisation problem. Many studies have considered diversity 1) as a mean to explore niches in behavioural space (quality diversity) or 2) to increase the structural differences of solutions (evolutionary diversity optimisation). In this study, we introduce a co-evolutionary algorithm to simultaneously explore the two spaces for the multi-component traveling thief problem. The results show the capability of the co-evolutionary algorithm to achieve significantly higher diversity compared to the baseline evolutionary diversity algorithms from the literature.

Keywords: Quality diversity · Co-evolutionary algorithms · Evolutionary diversity optimisation · Traveling thief problem

1 Introduction

Diversity has gained increasing attention in the evolutionary computation community in recent years. In classical optimisation problems, researchers seek a single solution that results in an optimal value for an objective function, generally subject to a set of constraints. The importance of having a diverse set of solutions has been highlighted in several studies [15,17]. Having such a set of solutions provides researchers with 1) invaluable information about the solution space, 2) robustness against imperfect modelling and minor changes in problems and 3) different alternatives to involve (personal) interests in decision-making. Traditionally, diversity is seen as exploring niches in the fitness space. However, two paradigms, namely quality diversity (QD) and evolutionary diversity optimisation (EDO), have been formed in recent years.

QD achieves diversity in exploring niches in behavioural space. QD maximises the quality of a set of solutions that differ in a few predefined features. Such a set of solutions can aid in the grasp of the high-quality solutions' behaviour in the feature space. QD has a root in novelty search, where researchers seek solutions with new behaviour without considering their quality [10]. For the first time,

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
G. Rudolph et al. (Eds.): PPSN 2022, LNCS 13398, pp. 237–249, 2022.
https://doi.org/10.1007/978-3-031-14714-2_17

a mechanism is introduced in [5] to keep best-performing solutions whereby, searching for unique behaviours. At the same time, the MAP-Elites framework was introduced in [4] to plot the distribution of high-performing solutions over a behavioural space. It has been shown that MAP-Elites is efficient in evolving behavioural repertoires. Later, the problem of computing a set of best-performing solutions differing in terms of some behavioural features is formulated and named QD in [22,23].

In contrast to QD, the goal of EDO is to explicitly maximise the structural diversity of a set of solutions that all have a desirable minimum quality. This approach was first introduced in [25] in the context of continuous optimisation. Later, EDO was adopted to generate images and benchmark instances for the traveling salesperson problem (TSP) [3,9]. Star-discrepancy and performance indicators from multi-objective evolutionary optimisation were adopted to achieve the same goals in [14,15]. In recent years EDO was studied in the context of well-known combinatorial optimisation problems, such as the quadratic assignment problem [7], the minimum spanning tree problem [2], the knapsack problem [1], and the optimisation of monotone sub-modular functions [13]. Distance-based diversity measures and entropy have been incorporated into EDO to evolve diverse sets of high-quality solutions for the TSP [6,18]. Nikfarjam et al. [17] introduced an EAX-based crossover focusing on structural diversification of TSP solutions. Most recently, Neumann et al. [12] introduced a co-evolutionary algorithm to find Pareto-front for bi-objective optimisation problem and simultaneously evolve another population to maximise structural diversity.

In this paper, we introduce a co-evolutionary algorithm (Co-EA) to compute two sets of solutions simultaneously; one employs the QD concept and the other evolves towards EDO. We consider the traveling thief problem (TTP) as a well-studied multi-component optimisation problem. QD and EDO have separately been studied in the context of TTP in [19] and [20], respectively. However, the Co-EA has several advantages:

- QD provides researchers with invaluable information about the distribution of best-performing solutions in behavioural space and enables decision-makers to select the best solution having their desirable behaviour. On the other hand, EDO provides us with robustness against imperfect modelling and minor changes in problems. We can benefit from both paradigms by using the Co-EA.
- Optimal or close-to optimal solutions are required in most EDO studies for initialization. The Co-EA eliminates this restriction.
- We expect the Co-EA brings about better results, especially in terms of structural diversity since the previous frameworks are built upon a single solution (the optimal solution). The Co-EA eliminates this drawback.
- The Co-EA benefits from a self-adaptation method to tune and adjust some hyper-parameters during the search improving the results meaningfully.

The remainder of the paper is structured as follows. We formally define the TTP and diversity in Sect. 2. The Co-EA is introduced in Sect. 3. We conduct comprehensive experimental investigation to evaluate Co-EA in Sect. 4. Finally, we finish with concluding remarks.

2 Preliminaries

In this section, we introduce the traveling thief problem and outline different diversity optimisation approaches established for this problem.

2.1 The Traveling Thief Problem

The traveling thief problem (TTP) is a multi-component combinatorial optimization problem. I. e., it is a combination of the classic traveling salesperson problem (TSP) and the knapsack problem (KP). The TSP is defined on a graph $G = (V, E)$ with a node set V of size n and a set of pairwise edges E between the nodes, respectively. Each edge, $e = (u, v) \in E$ is associated with a non-negative distance $d(e)$. In the TSP, the objective is to compute a tour/permutation $x : V \to V$ which minimizes the objective function

$$f(x) = d(x(n), x(1)) + \sum_{i=1}^{n-1} d(x(i), x(i+1)).$$

The KP is defined on a set of items I with $m := |I|$. Each item $i \in I$ has a profit p_i and a weight w_i. The goal is to determine a selection of items, in the following encoded as a binary vector $y = (y_1, \ldots, y_m) \in \{0, 1\}$, that maximises the profit, while the selected items' total weight does not exceed the capacity $W > 0$ of the knapsack:

$$g(y) = \sum_{j=1}^{m} p_j y_j \text{ s. t. } \sum_{j=1}^{m} w_j y_j \leq W.$$

Here, $y_j = 1$ if the jth item is included in the selection and $y_j = 0$ otherwise.

The TTP is defined on a graph G and a set of items I. Each node i except the first one includes a set of items $M_i \subseteq I$. In TTP, a thief visits each city exactly once and picks some items into the knapsack. A rent R is to be paid for the knapsack per time unit, and the speed of thief non-linearly depends on the weight W_{x_i} of selected items so far. Here, the objective is to find a solution $p = (x, y)$ including a tour x and a packing list (the selection of items) y that maximises the following function subject to the knapsack capacity:

$$z(p) = g(y) - R\left(\frac{d(x(n), x(1))}{\nu_{\max} - \nu W_{x_n}} + \sum_{i=1}^{n-1} \frac{d(x(i), x(i+1))}{\nu_{\max} - \nu W_{x_i}}\right) \text{ s. t. } \sum_{j=1}^{m} w_j y_j \leq W.$$

where $\nu_{\max}$ and $\nu_{\min}$ are the maximal and minimal traveling speed, and $\nu = \frac{\nu_{\max} - \nu_{\min}}{W}$.

2.2 Diversity Optimisation

This study simultaneously investigates QD and EDO in the context of the TTP. For this purpose, two populations P_1 and P_2 co-evolve. P_1 explores niches in

the behavioural space and the P_2 maximises its structural diversity subject to a quality constraint. In QD, a behavioural descriptor (BD) is defined to determine to which part of the behavioural space a solution belongs. In line with [19], we consider the length of tours $f(x)$, and the profit of selected items $g(y)$, to serve as the BD. To explore niches in the behavioural space, we propose a MAP-Elites-based approach in the next section.

For maximising structural diversity, we first require a measure to determine the diversity. For this purpose, we employ the entropy-based diversity measure in [20]. Let $E(P_2)$ and $I(P_2)$ denote the set of edges and items included in population P_2. The structural entropy of P_2 defines on two segments, the frequency of edges and items included in $E(P_2)$ and $I(P_2)$, respectively. let name these two segments edge and item entropy and denote them by H_e and H_i. H_e and H_i are calculated as

$$H_e(P_2) = \sum_{e \in E(P_2)} h(e) \text{ with } h(e) = -\left(\frac{f(e)}{\sum_{e \in I} f(e)}\right) \cdot \ln\left(\frac{f(e)}{\sum_{e \in I} f(e)}\right)$$

and

$$H_i(P_2) = \sum_{i \in I(P_2)} h(i) \text{ with } h(i) = -\left(\frac{f(i)}{\sum_{i \in I} f(i)}\right) \cdot \ln\left(\frac{f(i)}{\sum_{i \in I} f(i)}\right)$$

where $h(e)$ and $h(i)$ denote the contribution of edge e and item i to the entropy of P_2, respectively. Also, the terms $f(e)$ and $f(i)$ encode the number of solutions in P_2 that include e and i. It has been shown that $\sum_{e \in E(P_2)} f(e) = 2n\mu$ in [18], where $\mu = |P_2|$, while the number of selected items in P_2 can fluctuate. The overall entropy of P_2 is calculated by summation

$$H(P_2) = H_e(P_2) + H_i(P_2).$$

P_2 evolves towards maximisation of $H(P_2)$ subject to $z(p) \geq z_{\min}$ for all $p \in P_2$. Overall, we maximise the solutions' quality and their diversity in the feature-space through P_1, while we utilise P_2 to maximises the structural diversity.

3 Co-evolutionary Algorithm

This section presents a co-evolutionary algorithm – outlined in Algorithm 1 – to simultaneously tackle QD and EDO problems in the context of TTP. The algorithm involves two populations P_1 and P_2, employing MAP-Elite-based and EDO-based selection procedures.

3.1 Parent Selection and Operators

A bi-level optimisation procedure is employed to generate offspring. A new tour is generated by crossover at the first level; then, $(1 + 1)$ EA is run to optimise the packing list for the tour. The crossover is the only bridge between P_1 and

P_2. For the first parent we first select P_1 or P_2 uniformly at random. Then, one individual, $p_1(x_1, y_1)$ is selected again uniformly at random from the chosen population; the same procedure is repeated for the selection of the second parent $p_2(x_2, y_2)$. To generate a new solution $p'(x', y')$ from $p_1(x_1, y_1)$ and $p_2(x_2, y_2)$, a new tour $x' \leftarrow crossover(x_1, x_2)$ is first generated by EAX-1AB crossover. Edge-assembly crossover (EAX) is a high-performing operator and yields strong results in solving TSP. Nikfarjam et al. [19] showed that the crossover performs decently for the TTP as well.

EAX-1AB includes three steps: It starts with generating a so-called AB-Cycle of edges by alternatively selecting the edges from parent one and parent two. Next, an intermediate solution is formed. Having the first parent's edges copied to the offspring, we delete parent one's edges included in the AB-cycle and add the rest of edges in the AB-cycle. In this stage, we can have either a complete tour or a number of sub-tours. In latter case, we connect all the sub-tours one by one stating from the sub-tour with minimum number of edges. For connecting two sub-tours, we discard one edge from each sub-tour and add two new edges, a 4-tuple of edges. The 4-tuple is selected by following local search by choosing

$$(e_1, e_2, e_3, e_4) = \arg\min\{-d(e_1) - d(e_2) + d(e_3) + d(e_4)\}.$$

Note that if $E(t)$ and $E(r)$ respectively show the set of edges of the intermediate solution t and the sub-tour r, $e_1 \in E(r)$ $e_2 \in E(t) \setminus E(r)$. We refer interested readers to [11] for details on the implementation of the crossover.

Then, an internal $(1 + 1)$ EA is started to optimise a packing list y' for the new tour x' and form a complete TTP solution $p'(x', y')$. The new solution first inherits the first parent's packing list, $y' \leftarrow y_1$. Next, a new packing list is generated by standard bit-flip mutation ($y'' \leftarrow mutation(y')$). If $z(x', y'') > z(x', y')$, the new packing list is replaced with old one, $y' \leftarrow y''$. These steps repeats until an internal termination criterion for the $(1 + 1)$ EA is met. The process of generating a new solution $p'(x', y')$ is complete here, and we can ascend to survival selection.

3.2 Survival Selection Procedures

In MAP-elites, solutions with similar BD compete, and usually, the best solution survives to the next generation. To formally define the similarity and tolerance of acceptable differences in BD, the behavioural space is split into a discrete grid, where each solution belongs to only one cell. Only the solution with the highest objective value is kept in a cell in survival selection. The map not only contributes to the grasp of the high-quality solutions' behaviour but also does maintain the diversity of the population and aids to avoid premature convergence.

In this study, we discretize the behavioural space in the same way [19] did. They claimed that it is beneficial for the computational costs if we focus on a promising portion of behavioural space. In TTP, solely solving either TSP or KP is insufficient to compute a high-quality TTP solution. However, a solution $p(x, y)$ should score fairly good in both $f(x)$ and $g(y)$ in order to result in a high

TTP value $z(p)$. Thus, we limit the behavioural space to the neighbourhood close to optimal/near-optimal values of the TSP and the KP sub-problems. In other words, a solution $p(x, y)$ should result in $f(x) \in [f^*, (1 + \alpha_1) \cdot f^*]$ and $g(y) \in [(1 - \alpha_2) \cdot g^*, g^*]$. Note that f^* and g^* are optimal/near-optimal values of the TSP and the KP sub-problems, and α_1 and α_2 are acceptable thresholds to f^* and g^*, respectively. We obtain f^* and g^* by EAX [11] and dynamic programming [24]. Next, We discretize the space into a grid of size $\delta_1 \times \delta_2$. Cell (i, j), $1 \leq i \leq \delta_1$, $1 \leq j \leq \delta_2$ contains the best solution, with

$$f(x) \in \left[f^* + (i-1) \cdot \left(\frac{\alpha_1 f^*}{\delta_1} \right), f^* + i \cdot \left(\frac{\alpha_1 f^*}{\delta_1} \right) \right]$$

and

$$g(y) \in \left[(1-\alpha_2) \cdot g^* + (j-1) \cdot \left(\frac{\alpha_2 g^*}{\delta_2} \right), (1-\alpha_2) \cdot g^* + j \cdot \left(\frac{\alpha_2 g^*}{\delta_2} \right) \right].$$

After generating a new solution $p'(x', y')$, we find the cell corresponding with its BD $(f(x'), g(y'))$; if the cell is empty, p' is added to the cell. Otherwise, the solution with highest TTP value is kept in the cell.

Having defined the survival selection of P_1, we now look at P_2's survival selection based on EDO. We add $p'(x', y')$ to P_2 if the quality criterion is met, i. e., $z(p') \geq z_{\min}$. If $|P_2| = \mu + 1$, a solution with the least contribution to $H(P)$ will be discarded.

3.3 Initialisation

Population P_1 only accepts solutions with fairly high BDs $(f(x), g(y))$, and there is a quality constraint for P_2. Random solutions are unlikely to have these characteristics. As mentioned, we use the GA in [11] to obtain f^*; since the GA is a population-based algorithm, we can derive the tours in the final population resulting in a fairly good TSP score. Afterwards, we run the $(1+1)$ EA described above to compute a high-quality packing list for the tours. These packing lists also bring about a high KP score that allows us to populate P_1. Depending on the quality constraint $z_{\min}$, the initial solutions may not meet the quality constraint. Thus, it is likely that we have to initialize the algorithm with only P_1 until the solutions comply with the quality constraint; then, we can start to populate P_2. Note that both parents are selected from P_1 while P_2 is still empty. We stress that in most previous EDO-studies an optimal (or near-optimal solution) was required to be known a-priori and for initialization. In the proposed Co-EA, this strong requirement is no longer necessary.

3.4 Self Adaptation

Generating offspring includes the internal $(1 + 1)$ EA to compute a high-quality packing list for the generated tour. In [19], the $(1 + 1)$ EA is terminated after a fixed number of $t = 2m$ fitness evaluations. However, improving the quality of

Algorithm 1. The Co-Evolutionary Diversity Algorithm

```
Find the optimal/near-optimal values of the TSP and the KP by algorithms in [11, 24], respectively.
Generate an empty map and populate it with the initialising procedure.
while termination criterion is not met do
    Select two individuals based on the parent selection procedure and generate offspring by EAX and (1 + 1) EA.
    if The offspring's TSP and the KP scores are within α1, and α2 thresholds to the optimal values of BD. then
        Find the corresponding cell to the TSP and the KP scores in the QD map.
        if The cell is empty then
            Store the offspring in the cell.
        else
            Compare the offspring and the individual occupying the cell and store the best individual in terms of TTP score in the cell.
    if The offspring complies with the quality criterion then
        Add the offspring to the EDO population.
        if The size of EDO population is equal to μ + 1 then
            Remove one individual from the EDO population with the least contribution to diversity.
```

solutions is easier in the beginning and gets more difficult as the search goes on. Thus, we adopt a similar self-adaptation method proposed in [8,16] to adjust t during the search. Let $Z = \arg\max_{p \in P_1}\{z(p)\}$. Success defines an increase in Z. We discretize the search to intervals of u fitness evaluations. An interval is successful if Z increases; otherwise it is a failure. We reset t after each interval; t decreases if Z increases during the last interval. Otherwise, t increases to give the internal $(1 + 1)$ EA more budget in the hope of finding better packing lists and better TTP solutions. Here, we set $t = \gamma m$ where γ can take any value in $[\gamma_{\min}, \gamma_{\max}]$. We set

$$\gamma := \max\{\gamma \cdot F_1, \gamma_{\min}\} \text{ and } \gamma := \min\{\gamma \cdot F_2, \gamma_{\max}\}$$

in case of success and failure respectively. In our experiments, we use $F_1 = 0.5$, $F_2 = 1.2$, $\gamma_{\min} = 1$, $\gamma_{\max} = 10$, and $u = 2000m$ based on preliminary experiments. We refer to this method as $Gamma_1$.

Moreover, we propose an alternative terminating criterion for the internal $(1 + 1)$ EA, and denote it $Gamma_2$. Instead of running the $(1 + 1)$ EA for $t = \gamma m$, we terminate $(1 + 1)$ EA when it fails in improving the packing list in $t' = \gamma' m$ consecutive fitness evaluations. γ' is updated in the same way as γ. Based on the preliminary experiments, we set $\gamma'_{\min}$ and $\gamma'_{\max}$ to 0.1 and 1, respectively.

4 Experimental Investigation

We empirically study the Co-EA in this section. We run the Co-EA on eighteen TTP instances from [21], the same instances are used in [19]. We first illustrate

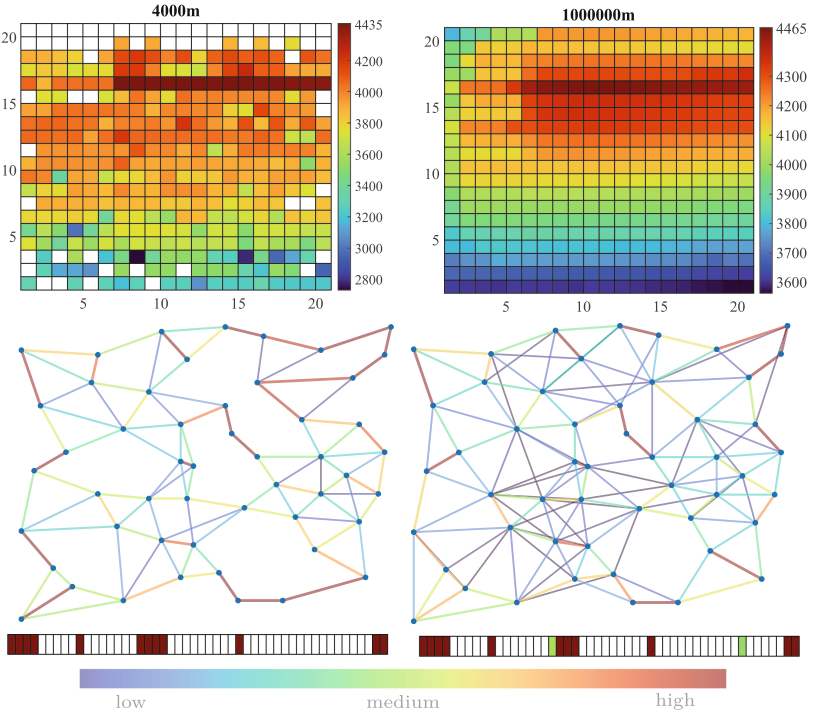

Fig. 1. Evolution of P_1 and P_2 over $4000m$ and $1000000m$ fitness evaluations on instance 1 with $\alpha = 2\%$. The first row depicts the distribution of high-quality solutions in the behavioural space (P_1). The second and the third rows show the overlay of all edges and items used in exemplary P_2, respectively. Edges and items are coloured by their frequency. (Color figure online)

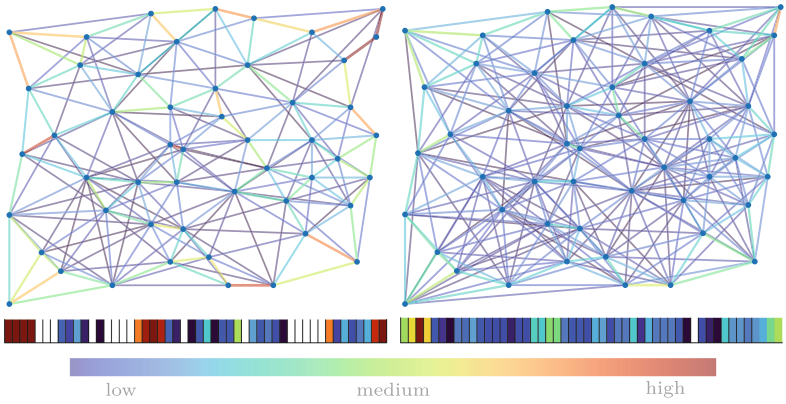

Fig. 2. Overlay of all edges and items used in an exemplary final population P_2 on instance 1 with $\alpha = 10\%$ (left) and $\alpha = 50\%$ (right). Edges and items are coloured by their frequency. (Color figure online)

the distribution of solutions in P_1, and the structural diversity of solutions in P_2. Then, we compare the self-adaptation methods with the fixed parameter setting. Afterwards, we conduct a comprehensive comparison between P_1 and P_2 and the populations obtained by [19] and [20]. Here, the termination criterion and α are set on $1000000m$ fitness evaluations and 10%, respectively.

MAP-Elite selection can be beneficial to illustrate the distribution of high-quality solutions in the behaviour space. On the other hand, EDO selection aims to understand which elements in high-quality solutions is easy/difficult to be replaced. Figure 1 depicts exemplary populations P_1 and P_2 after $4000m$ and $1000000m$ fitness evaluations of Co-EA on instance 1, where $\alpha = 2\%$. The first row illustrates the distribution P_1's high-performing solutions over the behavioural space of $f(x)$ and $g(y)$. The second and the third rows represent the overlay of edges and items in P_2, respectively. The figure shows the solutions with highest quality are located on top-right of the map on this test instance where the gaps of $f(x)$ and $g(y)$ to f^* and g^* are in [0.0150.035] and [0.150.18], respectively. In the second row of the figure, we can observe that Co-EA successfully incorporates new edges into P_2 and reduces the edges' frequency within the population. However, it is unsuccessful in incorporating new items in P_2. The reason can be that there is a strong correlation between items in this particular test instance, and the difference in the weight and profit of items is significant. It means that there is not many other good items to be replaced with the current selection. Thus, we cannot change the items easily when the quality criterion is fairly tight ($\alpha = 2\%$). As shown on the third row of the figure, the algorithm can change i_8 with i_{43} in some packing lists.

Figure 2 reveals that, as α increases, so does the room to involve more items and edges in P_2. In other words, there can be found more edges and items to be included in P_2. Figure 2 shows the overlays on the same instances, where α is set to 10% (left) and 50% (right). Not only more edges and items are included in P_2 with the increase of α, but also Co-EA reduces the frequency of the edges and items in P_2 to such a degree that we can barely see any high-frequent edges or items in the figures associated with $\alpha = 50\%$. Moreover, the algorithm can successfully include almost all items in P_2 except item i_{39}. Checking the item's weight, we notice that it is impossible to incorporate the item into any solution. This is because, $w_{i_{39}} = 4\,400$, while the capacity of the knapsack is set to $4\,029$. In other words, $w_{i_{39}} > W$.

4.1 Analysis of Self-Adaptation

In this sub-section, we compare the two proposed termination criteria and self-adaptation methods $Gamma_1$ and $Gamma_2$ with the fixed method employed in [19]. We incorporate these methods into the Co-EA and run it for ten independent runs. Table 1 summarises the mean of P_2's entropy obtained from the competitors. The table indicates that both $Gamma_1$ and $Gamma_2$ outperform the fixed method on all test instances. Kruskal-Wallis statistical tests at significance level 5% and Bonferroni correction also confirm a meaningful difference in median of results for all instances except instance 15 where there is no significant

Table 1. Comparison of $Gamma_1$ (1) and $Gamma_2$ (2), and the fixed method (3). The instances are numbered as Table 1 in [19]. In columns Stat the notation X^+ means the median of the measure is better than the one for variant X, X^- means it is worse, and X^* indicates no significant difference. Stat shows the results of Kruskal-Wallis statistical test at a significance level of 5% and Bonferroni correction. Also, $p^* = \max_{p \in P_1}\{z(p)\}$.

	$H(P_2)$						$z(p^*)$					
Inst.	$Gamma_1$	(1)	$Gamma_2$	(2)	$fixed$	(3)	$Gamma_1$	(1)	$Gamma_2$	(2)	$fixed$	(3)
	mean	Stat	mean	Stat	mean	Stat	mean	Stat	mean	Stat	mean	Stat
1	8.7	2^-3^+	8.8	1^+3^+	8.2	1^-2^-	4452.4	2^-3^*	4465	1^+3^*	4461.1	1^*2^*
2	9.3	2^*3^+	9.3	1^*3^+	9.1	1^-2^-	8270.4	2^*3^*	8232.2	1^*3^*	8225.2	1^*2^*
3	9.9	2^+3^+	9.8	1^-3^+	9.6	1^-2^-	13545.4	2^*3^*	13607.5	1^*3^*	13609	1^*2^*
4	7.7	2^-3^+	7.7	1^+3^+	7.4	1^-2^-	1607.1	2^*3^*	1607.5	1^*3^*	1607.5	1^*2^*
5	9	2^*3^+	9	1^*3^+	8.8	1^-2^-	4814.7	2^*3^*	4805.3	1^*3^*	4811	1^*2^*
6	9.4	2^*3^+	9.4	1^*3^+	9.2	1^-2^-	6834.5	2^*3^*	6850	1^*3^*	6850	1^*2^*
7	8	2^*3^+	8.1	1^*3^+	7.6	1^-2^-	3200.8	2^*3^*	3218.4	1^*3^*	3165	1^*2^*
8	9	2^-3^+	9	1^+3^+	8.8	1^-2^-	7854.2	2^*3^*	7854.2	1^*3^*	7850.9	1^*2^*
9	9.5	2^*3^+	9.5	1^*3^+	9.3	1^-2^-	13644.8	2^+3^+	13644.8	1^-3^+	13644.8	1^-2^-
10	10.5	2^-3^+	10.5	1^+3^+	10.1	1^-2^-	11113.6	2^*3^*	11145.7	1^*3^-	11148	1^*2^+
11	11.2	2^*3^+	11.2	1^*3^+	11	1^-2^-	25384.6	2^+3^*	25416.6	1^-3^-	25401.3	1^*2^+
12	9.3	2^*3^+	9.4	1^*3^+	9.2	1^-2^-	3538.2	2^*3^*	3564.4	1^*3^*	3489.4	1^*2^*
13	10.7	2^*3^+	10.7	1^*3^+	10.5	1^-2^-	13369.3	2^+3^*	13310.4	1^-3^*	13338.4	1^*2^*
14	7.7	2^-3^*	9.7	1^+3^+	8.5	1^*2^-	5261.9	2^*3^*	5410.3	1^*3^*	5367.1	1^*2^*
15	10.9	2^*3^+	10.9	1^*3^+	10.7	1^-2^-	20506.8	2^*3^*	20506.8	1^*3^*	20385.3	1^*2^*
16	11.6	2^-3^*	11.7	1^+3^+	11.4	1^*2^-	18622.2	2^*3^*	18609.6	1^*3^*	18641.4	1^*2^*
17	11.2	2^*3^+	11.2	1^*3^+	11.1	1^-2^-	9403.8	2^*3^*	9448.3	1^*3^*	9428.1	1^*2^*
18	11.4	2^*3^+	11.4	1^*3^+	11.1	1^-2^-	19855.3	2^-3^*	19943.8	1^+3^*	19879.3	1^*2^*

difference in the mean of $Gamma_1$ and the fixed method. In comparison between $Gamma_1$ and $Gamma_2$, the latter outperforms the first in 4 test instances, while it is surpassed in only one case. In conclusion, Table 1 indicates that $Gamma_2$ works the best with respect to the entropy of P_2.

Moreover, Table 1 also shows the mean TTP score of the best solution in P_1 obtained from the three competitors. Although Table 1 indicates that the statistical test cannot confirm a significant difference in the mean of the best TTP solutions, $Gamma_2$'s results are slightly better in 7 cases, while $Gamma_1$ and $fixed$ have better results in 3 cases. Overall, all three competitors perform almost equally in terms of the best TTP score. Since $Gamma_2$ outperforms other methods in entropy, we employ it for the Co-EA in the rest of the study.

4.2 Analysis of Co-EA

This section compares P_1 and P_2 with the QD-based EA in [19] and the standard EDO algorithm, respectively. Table 2 summarises this series of experiments. The results indicate that the Co-EA outperforms the standard EDO in 14 instances, while the EDO algorithm has a higher entropy average in only two cases. In the two other test instances, both algorithms performed equally. Moreover, the Co-EA yields competitive results in terms of the quality of the best solution

Table 2. Comparison of the Co-EA and QD from [19] in terms of $z(p^*)$, and EDO algorithm from [20] in $H(P_2)$. Stat shows the results of Mann-Whitney U-test at significance level 5%. The notations are in line with Table 1.

Inst.	Co-EA	(1)	QD	(2)	Co-EA	(1)	EDO	(2)
	Q	Stat	Q	Stat	H	Stat	H	Stat
1	4465	2^*	4463.5	1^*	8.8	2^*	8.6	1^*
2	8232.2	2^*	8225.7	1^*	9.3	2^-	9.4	1^+
3	13607.5	2^*	13544.9	1^*	9.8	2^-	9.8	1^+
4	1607.5	2^+	1607.5	1^-	7.7	2^+	7.7	1^-
5	4805.3	2^*	4813.2	1^*	9	2^*	9	1^*
6	6850	2^*	6806.8	1^*	9.4	2^+	9.3	1^-
7	3218.4	2^*	3191.9	1^*	8.1	2^+	8	1^-
8	7854.2	2^*	7850.9	1^*	9	2^*	9	1^*
9	13644.8	2^+	13644.8	1^-	9.5	2^*	9.5	1^*
10	11145.7	2^-	11149.2	1^+	10.5	2^+	10.2	1^-
11	25416.6	2^-	25555.2	1^+	11.2	2^+	11	1^-
12	3564.4	2^*	3514	1^*	9.4	2^+	8.8	1^-
13	13310.4	2^*	13338.6	1^*	10.7	2^+	10.2	1^-
14	5410.3	2^*	5364.6	1^*	9.7	2^+	9.5	1^-
15	20506.8	2^*	20499.2	1^*	10.9	2^+	10.7	1^-
16	18609.6	2^-	18666.4	1^+	11.7	2^+	11.1	1^-
17	9448.3	2^*	9407.7	1^*	11.2	2^+	10.4	1^-
18	19943.8	2^+	19861.8	1^-	11.4	2^+	11.1	1^-

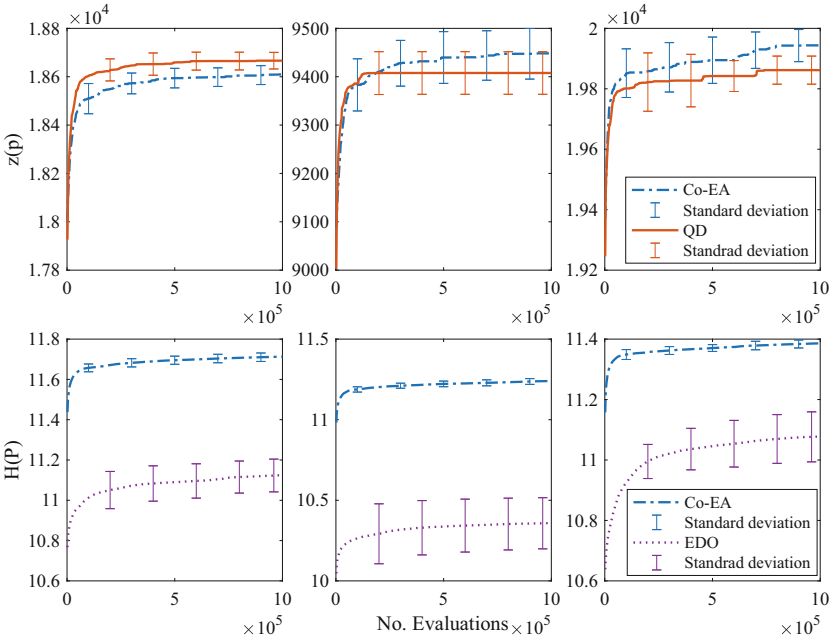

Fig. 3. Representative trajectories of Co-EA and standard EDO EA on instances 16, 17, 18. The top row shows $H(p_2)$ while the second row shows the best solution in P_1.

compared to the QD-based EA; in fact, the Co-EA results in a higher mean of TTP scores on 12 test instances. For example, the best solutions found by Co-EA score 19943.8 on average, whereby the figure stands at 19861.8 for the QD-based algorithms.

Figure 3 depicts the trajectories of Co-EA and the standard EDO algorithm in entropy of the population (the first row), and that of Co-EA and QD-based EA in quality of the best solution (the second row). Note that in the first row the x-axis shows fitness evaluations from $4000m$ to $1000000m$. This is because P_2 is empty in the early stages of running Co-EA and we cannot calculate the entropy of P_2 until $|P_2| = \mu$ for the sake of fair comparison. The figure shows that Co-EA converges faster and to a higher entropy than the standard EDO algorithm. Moreover, it also depicts results obtained by Co-EA has much less standard deviation. Regarding the quality of the best solution, both Co-EA and QD-based EA follow a similar trend.

5 Conclusion

We introduced a co-evolutionary algorithm to simultaneously evolve two populations for the traveling thief problem. The first population explore niches in a behavioural space and the other maximises structural diversity. The results showed superiority of the algorithm to the standard framework in the literature in maximising diversity. The co-evolutionary algorithm also yields competitive results in terms of quality.

It is intriguing to adopt more complicated MAP-Elites-based survival selection for exploring the behavioural space. Moreover, this study can be a transition from benchmark problems to real-world optimisation problems where imperfect modelling is common and diversity in solutions can be beneficial.

Acknowledgements. This work was supported by the Australian Research Council through grants DP190103894 and FT200100536.

References

1. Bossek, J., Neumann, A., Neumann, F.: Breeding diverse packings for the knapsack problem by means of diversity-tailored evolutionary algorithms. In: GECCO, pp. 556–564. ACM (2021)
2. Bossek, J., Neumann, F.: Evolutionary diversity optimization and the minimum spanning tree problem. In: GECCO, pp. 198–206. ACM (2021)
3. Chagas, J.B.C., Wagner, M.: A weighted-sum method for solving the bi-objective traveling thief problem. CoRR abs/2011.05081 (2020)
4. Clune, J., Mouret, J., Lipson, H.: Summary of "the evolutionary origins of modularity". In: GECCO (Companion), pp. 23–24. ACM (2013)
5. Cully, A., Mouret, J.: Behavioral repertoire learning in robotics. In: GECCO, pp. 175–182. ACM (2013)
6. Do, A.V., Bossek, J., Neumann, A., Neumann, F.: Evolving diverse sets of tours for the travelling salesperson problem. In: GECCO, pp. 681–689. ACM (2020)

7. Do, A.V., Guo, M., Neumann, A., Neumann, F.: Analysis of evolutionary diversity optimisation for permutation problems. In: GECCO, pp. 574–582. ACM (2021)
8. Doerr, B., Doerr, C.: Optimal parameter choices through self-adjustment: applying the 1/5-th rule in discrete settings. In: GECCO Companion, pp. 1335–1342 (2015)
9. Gao, W., Nallaperuma, S., Neumann, F.: Feature-based diversity optimization for problem instance classification. Evol. Comput. **29**(1), 107–128 (2021)
10. Lehman, J., Stanley, K.O.: Abandoning objectives: evolution through the search for novelty alone. Evol. Comput. **19**(2), 189–223 (2011)
11. Nagata, Y., Kobayashi, S.: A powerful genetic algorithm using edge assembly crossover for the traveling salesman problem. INFORMS J. Comput. **25**(2), 346–363 (2013)
12. Neumann, A., Antipov, D., Neumann, F.: Coevolutionary Pareto diversity optimization. CoRR arXiv:2204.05457 (2022), accepted as full paper at GECCO 2022
13. Neumann, A., Bossek, J., Neumann, F.: Diversifying greedy sampling and evolutionary diversity optimisation for constrained monotone submodular functions. In: GECCO, pp. 261–269. ACM (2021)
14. Neumann, A., Gao, W., Doerr, C., Neumann, F., Wagner, M.: Discrepancy-based evolutionary diversity optimization. In: GECCO, pp. 991–998. ACM (2018)
15. Neumann, A., Gao, W., Wagner, M., Neumann, F.: Evolutionary diversity optimization using multi-objective indicators. In: GECCO, pp. 837–845. ACM (2019)
16. Neumann, A., Szpak, Z.L., Chojnacki, W., Neumann, F.: Evolutionary image composition using feature covariance matrices. In: GECCO, pp. 817–824 (2017)
17. Nikfarjam, A., Bossek, J., Neumann, A., Neumann, F.: Computing diverse sets of high quality TSP tours by EAX-based evolutionary diversity optimisation. In: FOGA, pp. 9:1–9:11. ACM (2021)
18. Nikfarjam, A., Bossek, J., Neumann, A., Neumann, F.: Entropy-based evolutionary diversity optimisation for the traveling salesperson problem. In: GECCO, pp. 600–608. ACM (2021)
19. Nikfarjam, A., Neumann, A., Neumann, F.: On the use of quality diversity algorithms for the traveling thief problem. CoRR abs/2112.08627 (2021)
20. Nikfarjam, A., Neumann, A., Neumann, F.: Evolutionary diversity optimisation for the traveling thief problem. CoRR abs/2204.02709 (2022)
21. Polyakovskiy, S., Bonyadi, M.R., Wagner, M., Michalewicz, Z., Neumann, F.: A comprehensive benchmark set and heuristics for the traveling thief problem. In: GECCO, pp. 477–484. ACM (2014)
22. Pugh, J.K., Soros, L.B., Stanley, K.O.: Quality diversity: a new frontier for evolutionary computation. Front. Robot. AI **3**, 40 (2016)
23. Pugh, J.K., Soros, L.B., Szerlip, P.A., Stanley, K.O.: Confronting the challenge of quality diversity. In: GECCO, pp. 967–974. ACM (2015)
24. Toth, P.: Dynamic programming algorithms for the zero-one knapsack problem. Computing **25**(1), 29–45 (1980)
25. Ulrich, T., Thiele, L.: Maximizing population diversity in single-objective optimization. In: GECCO, pp. 641–648. ACM (2011)