Partition Crossover Evolutionary Algorithm for the Team Orienteering Problem with Time Windows

Ibtihel Ghobber(✉), Takwa Tlili, and Saoussen Krichen

LARODEC Laboratory, Institut Supérieur de Gestion Tunis,
Université de Tunis, Tunis, Tunisia
takwa.tlili@gmail.com

Abstract. The rapid evolution in tourism domain and new technologies make the search for the destination and site information for the tourists very difficult. In operations research field, this problem is modeled as Tourist Trip Design Problem (TTDP) which is about finding an optimal path-planning solution for tourists in order to visit multiple Points Of Interests (POIs). This paper addresses the team orienteering problem with time windows (TOPTW) that is an extension of TTDP. We apply for the first time the evolutionary algorithm based on partition crossover (EAPX) for solving the TOPTW. This approach is tested using a set of benchmarks then is compared to state-of-the-art algorithms to evaluate its performance. The results indicate the effectiveness of this method in solving the TOPTW.

Keywords: Team orienteering problem with time windows
Tourist Trip Design Problem · Partition crossover
Evolutionary algorithm · Meta-heuristics

1 Introduction

Tourists are often confused about selecting the interesting places to visit during their tour. Since there is a limitation in time and budget, tourists will select those destinations that have the more attractive Points of Interest (POIs). After discovering the interesting POIs, tourists decide to determine how many of them they can visit which path to follow, that fit their visiting time and travel budget limitations. In operations research, seeking for POIs and proposing a tour plan for tourists is modeled as Tourist Trip Design Problem (TTDP) that have been studied by numerous researchers. The main objective of this problem is finding the most interesting POIs that maximize tourist satisfaction, by incorporating visiting time limitations, and opening and closing days/hours of each POI. Gavalas et al. (2014) classified TTDP into two variants. The first is the single tour TTDP variants that aim to find a single path between nodes that maximizes the profit and minimizes the travel cost while the second variants

M. Mouhoub et al. (Eds.): IEA/AIE 2018, LNAI 10868, pp. 200–211, 2018.
https://doi.org/10.1007/978-3-319-92058-0_19

are the multiple tour TTDP which are characterized by determining a multiple POIs that the tourists need to visit with taking into account their visiting time. We focus on an extension of multiple tour TTDP which is the Team Orienteering Problem with Time Windows (TOPTW). This problem aims to find the optimal trip that respects tourists travel time and POIs' time windows and maximize their satisfaction which corresponds to maximize the total collected profit from each POI. Since the TOPTW is demonstrated as an NP-hard problem Vansteenwegen et al. (2009a) many meta-heuristic approaches are proposed in the literature to tackle it. Montemanni and Gambardella (2009) develop ant colony optimization (ACO) for solving TOPTW. This approach integrates ant colony system (ACS) algorithms which are based on a computational paradigm inspired by real ant colonies. It has two elements as follows. The construction phase in which feasible solutions are produced. The local search algorithm used to take down each solution generated in the construction phase to a local optimum. A very fast iterated local search (ILS) meta-heuristic is introduced by Vansteenwegen et al. (2009a) to deal with the TOPTW instances where it based on two steps as follows. The first is the insertion step attempts to add new visits to a tour. The second is the shake step is used to liberate local optima. The total average gap is only 1.8% compared to the ACS.

The main contribution of this paper is to present an evolutionary algorithm based on the partition crossover technique for solving the TOPTW. As noted that this algorithm has proved its performance in solving routing optimization problem. Herein, we detail its different steps and give its pseudo code. To validate the proposed model, we test a set of TOPTW benchmark instances. A comparative study is held to show the competitiveness of the proposed approach.

This paper is structured as follows. In Sect. 2, we define the TOPTW statement and propose its mathematical formulation. The proposed evolutionary algorithm based on partition crossover (EAPX) for TOPTW is explained in Sect. 3. In Sect. 4, computational experiments are provided. Section 5 summarizes the paper.

2 Problem Description

In e-tourism, the TOPTW is characterized by a set of POIs, where each point is associated with a profit which denotes its importance value for the tourist, a visit duration and a time window, as well as a travel time between each pair of POIs. The main goal this problem is to find a fixed number of disjoint routes from the starting node to the ending POI that respect a set of constraints.

Summarizing, the objective function of the TOPTW (1) seeks to maximize the total collected profit from each visited POI which corresponds to the maximization of tourist's ratings, depends on their preferences and considers the following constraints.

- The trip starts and ends at the hotel.
- Every POI is visited at most once.
- The total trip time is limited by $Tmax$.
- Each visit starts with a POI's time windows $[O_i, C_i]$.

2.1 Mathematical Formulation

The TOPTW involves a set of POIs, where each POI is associated with a score S_i a service time T_i and a time window $[O_i, C_i]$. For the depot vertex, which corresponds to the hotel e0 represents O_0 represents the the starting time to depart from the hotel and C_0 the arrival to it, as shown in the Fig. 1. This problem can be modeled as a graph $G = (V, A)$, where $V = \{1, \ldots, n\}$ is the set of POIs and $A = \{(i, j)|i, j \in V, i \neq j\}$ is the set of arcs that connect the POIs. Given m the number of tours. The starting location POI 1 and the end destination POI n of every tour are fixed. The travel time from POI i to POI j is denoted by $t_i j$. In this problem, not all locations can be visited since the total travel time is limited by a given time budget $Tmax$ and each POI can be visited at most once.

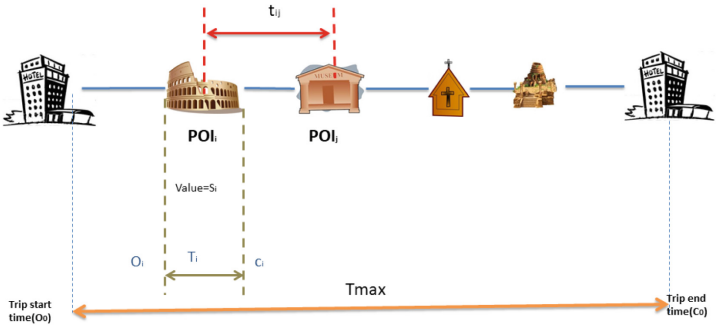

Fig. 1. The TOPTW illustration

Furthermore, the TOPTW uses the following parameters and decision variables.

The TOPTW objective function aims to determine m routes of maximum total collected scores (1).

$$Max\, Z(X) = \sum_{d=1}^{m} \sum_{i=2}^{n-1} S_i\, y_{id} \tag{1}$$

A number of constraints to be respected in the TOPTW can be as stated by (Vansteenwegen et al. 2009b; Cura 2014). These constraints are either (1) Routing constraints or (2) Time constraints detailed as follow.

Sets and parameters	
V	The total number of POIs
m	The number of tours
S_i	The score of POI i
T_i	The service time of POI i
$[O_i, C_i]$	The time window of POI i: The visit of POI i can only start during this time window
t_{ij}	The travel time from POI i to POI j
u_{id}	The position of location i in tour d
α_{ir}	The departure time from the location, which is at position r of tour d
$Tmax$	The maximum total travel time
Decision variables	
$x_{ijd} = \begin{cases} 1 & \text{if in path } d, \text{a visit to vertex } i \text{ is followed by a visit to vertex } j \\ 0 & \text{otherwise} \end{cases}$	
$y_{id} = \begin{cases} 1 & \text{if vertex } i \text{ is visited in path } d \\ 0 & \text{otherwise} \end{cases}$	

1. **Routing constraints**

$$\sum_{d=1}^{m}\sum_{j=2}^{n-1} x_{1jd} = \sum_{d=1}^{m}\sum_{i=2}^{n-1} x_{ind} = m \tag{2}$$

$$\sum_{d=1}^{m} y_{dk} \leq 1, k = 2, \ldots, n-1 \tag{3}$$

$$\sum_{i=1}^{n-1} x_{ikd} = \sum_{j=2}^{n-1} x_{kjd} = y_{kd}, k = 2, \ldots, n-1, d = 1, \ldots, m \tag{4}$$

$$2 \leq u_{id} \leq n, i = 2, \ldots, n; d = 1, \ldots, m \tag{5}$$

$$u_{id} - u_{jd} + 1 \leq (n-1)(1 - x_{ijd}), i = 2, \ldots, n; d = 1, \ldots, m \tag{6}$$

While the TOPTW can be considered as a vehicle routing problem, a set of routing requirements are taken into account. The constraints (2) guarantee that each tour starts from location 1 and ends at location n. Each POI is visited only once, ensured by Eq. (3). The continuity in a tour is needed in constraints (4) which guarantee that if a POI is visited in a given tour, it is preceded and followed by exactly one other POI in the same tour. The last routing constraints (5) and (6) are about preventing subtours.

2. **Time constraints**

$$\sum_{i=1}^{n-1}\left(T_i y_{id} + \sum_{j=2}^{n} t_{ij} x_{ijd}\right) \leq Tmax, d = 1, \ldots, m \alpha_{1d} = T_1 \tag{7}$$

$$\alpha_{1d} = T_1, d = 1, \ldots, m \tag{8}$$

$$\alpha_{rd} = max\left[\left(\alpha_{(r-1)d} + \sum_{i=2}^{n}\sum_{j=1}^{n-1}\begin{cases} t_{ij}x_{jid}y_{id} & \text{if } u_{id} = r \\ 0 & \text{otherwise} \end{cases}\right)\sum_{i=1}^{n}\begin{cases} O_i y_{id} & \text{if } u_{id} = r \\ 0 & \text{otherwise} \end{cases}\right] \tag{9}$$

$$+\sum_{i=1}^{n}\begin{cases} T_i y_{id} & \text{if } u_{id} = r \\ 0 & \text{otherwise} \end{cases}, r = 2, \ldots, n; d = 1, \ldots, m \tag{10}$$

$$C_i \geq \sum_{d=1}^{m}\sum_{r=2}^{n}\begin{cases} y_{id}(\alpha_{rd} - T_i) & \text{if } u_{id} = r \\ 0 & \text{otherwise} \end{cases}, i = 1, \ldots, n \tag{11}$$

The TOPTW's time restrictions are described as the constraints (7) which guarantee that each tour is completed within a given time limit. The departure time from the first POI is calculated as shown in the constraint (8), moreover, the departure time from the POI r of tour d is given by the Eq. (10). Finally, the time window is defined by Eq. (11).

3 Evolutionary Algorithm Based on Partition Crossover (EAPX)

To solve the TOPTW we propose a new approach named EAPX. It includes the following three main steps. The first phase is the initial population used for the construction the initial solution while the second phase is a local search method used for the improvement of the solutions found in the first phase and partition crossover method for the production of the new generation. All these phases are more detailed in the following subsections. A flowchart depicting the proposed meta-heuristic is given in the Fig. 2.

3.1 Initial Population

The algorithm starts with generating the initial solution Sol_0 using of the both the petal algorithm (PA) and diversity. The petal is designed for building the first half of the initial population. Its description is presented by Ryan et al. (1993), where the nodes are radially numbered about the depot and each petal is formed by a list of radially successive nodes. A feasibility of the petal is proved only if the total travel time does not exceed the imposed time limit T_{max}, which corresponds to latest arrival time to the depot and the nodes' time windows not violated. In this case, we propose a random order of node for each petal. Moreover, the second half of the initial solution is produced by diversity. Another solution is built from each individual of the initial solution with considering a random number of nodes swaps in each sub-route. Our population is formed

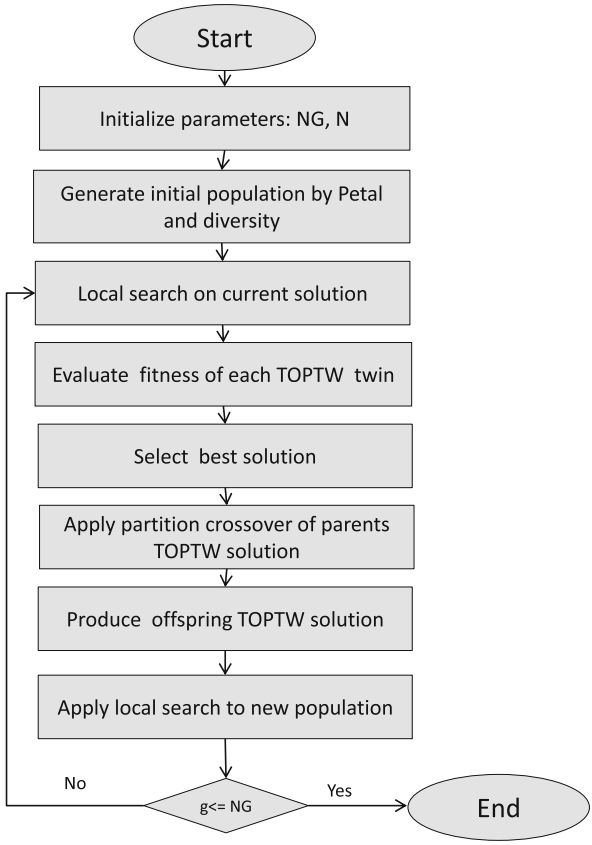

Fig. 2. A flowchart for the proposed EAPX meta-heuristic.

by two TOPTW solutions, constitute the twins. Therefore, each twin includes different sub-routes, which integrates a number of homogeneous petals. In order to find locally optimal solutions, we implement the 2-OPT operator method, used for decreasing the tour travel cost. This technique works on tour crossing over itself and reordered to find local optima tour.

3.2 Local Search

Given the initial population generated by the petal and diversity, a local search algorithm is proposed to improve the performance of the solution S. In this method, we implement many operations which, are detailed as follows. It starts with an arbitrary solution to the TOPTW then attempts to find a better solution by incrementally replacing two scheduled nodes within a route this process is done by the Swap operator. In this case, different combinations are processed for selecting the best nodes. The fitness value is calculated by the total profit. If the change produces a better solution, an incremental change is made to the new

solution, repeating until finding the solution that maximizes the fitness of each individual with no further improvements can be found. According to Whitley et al. (2009), the recombination of two local optimum solutions by the PX results usually in local optima produced child.

3.3 Partition Crossover for TOPTW

The reproductive process of our proposed approach is done by the partition crossover (PX) operator. The basic idea of this procedure is to recombine the local optima solutions with considering that each TOPTW twin of the produced solution is divided into a number of subroutes formed by the same nodes for each subtour. In this case, for each subtour, the order of the applied procedure can be explained as follows. First, the two parent solutions are chosen from the TOPTW solutions for constructing the graph G. Second, the constructed graph G is partitioned into subgraphs by deleting the common arcs between these two solutions. The identification of the linked components is executed by the Breadth first algorithm. Finally, the two offsprings are produced from the two parent solutions by recombining them.

3.4 Illustrative Example

For the problem with 23 POIs and $m = 3$ tours a sample solution is shown in the Fig. 3. This example is the result of the local search operator applied after generating the initial population where we present an undirected graph G with 23 POIs and the starting point labeled by 0. The first twin presents the solution 1 by the dashed red line the solution 2 corresponds to the second

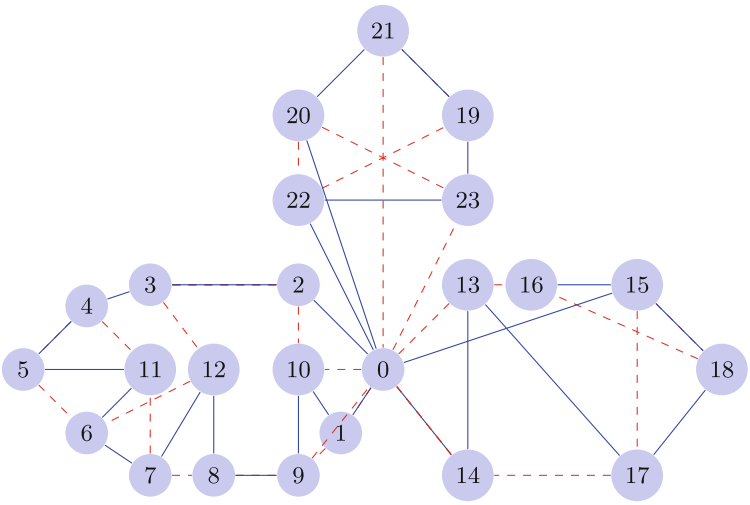

Fig. 3. An illustrative example including 23 POIs and 3 tours (Color figure online)

solution is shown by the solid blue line. In this case, the routes for the solution 1 in each tour are respectively $m1 = \{0, 1, 9, 8, 7, 11, 4, 5, 6, 12, 3, 2, 10, 0\}$, $m2 = \{0, 14, 17, 15, 18, 16, 13, 0\}$, and $m3 = \{0, 23, 20, 22, 19, 21, 0\}$ while for the solution 2 are presented as follows: $m1 = \{0, 1, 10, 9, 8, 12, 7, 6, 11, 5, 4, 3, 2, 0\}$, $m2 = \{0, 14, 13, 17, 18, 15, 0\}$, and $m3 = \{0, 22, 23, 19, 21, 20, 0\}$. After applying the partition crossover method the new generated offspring are showed in the Fig. 4 where the new routes of the first constructed offspring are respectively $m1 = \{0, 1, 10, 9, 8, 7, 11, 4, 5, 6, 12, 3, 2, 0\}$, $m2 = \{0, 14, 17, 15, 18, 16, 13, 0\}$, and $m3 = \{0, 23, 20, 22, 19, 21, 0\}$. The routes of each tour for the second offspring are detailed as follows: $m1 = \{0, 1, 10, 9, 8, 7, 11, 4, 55, 6, 12, 3, 2, 0\}$, $m2 = \{0, 14, 13, 17, 18, 15, 0\}$, and $m3 = \{0, 22, 23, 19, 21, 20, 0\}$.

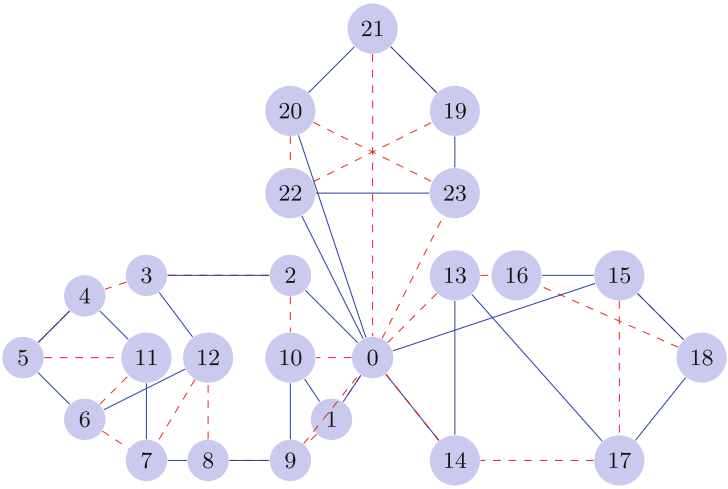

Fig. 4. A TOPTW solution presentation after applying PX (Color figure online)

4 Computational Experiments

The experimentation is designed to evaluate the performance of the EAPX in solving the TOPTW, using the benchmark instances. We first describe the TOPTW benchmark dataset. Then, we detail the results obtained from the experimentation and assess the effectiveness of the algorithm using a comparative study.

4.1 Experimental Setup

The EAPX was coded in Java language and running Windows 10. The experiments were run on a personal computer Hp equipped with Intel Pentium, 1,90 GHz CPU and 4 GB of RAM. In this case, for each instance of the benchmark, the EAPX is executed for 30 runs where its execution ends with best

TOPTW solutions and there is no improvement to be added. The parameters of the proposed algorithm are the number of generation NG = 50 and the number of individuals in half of the population N = 50.

4.2 Results and Discussion

In order to evaluate the performance of the EAPX approach in solving TOPTW, the obtained results are compared to the following methods.

(1) Iterated Local Search (ILS) (Vansteenwegen et al. 2009a) and
(2) Granular Variable Neighborhood Search Based on Linear Programming (GVNS) (Labadie et al. 2012).
(3) Artificial Bee Colony (ABC) (Cura 2014).

Tables 1 and 2 summarize respectively the results of EAPX and the comparison with ILS, GVNS, ABC, and GRASP-ELS for Solomon instances and Cordeau instances. In these tables, the first column describes the instance name, the second column contains the number of tours, and the third column gives the best-known solution (BKS) for the TOPTW. The following columns report respectively for each algorithm the value of the best profit, and the Gap (Sect. 4.2).

$$Gap = \frac{BKS - EAPX}{BKS} * 100\%$$

As seen in the Table 1 EAPX reaches the BKS in the following instances: c-100, r101, r102, r105 and rc107. The average gap between EAPX result and the best-known solution is only 1.39%. For 13 instances of Solomon the optimal solution of EAPX is known and for the other instances are close to the best known. Compared to ILS, GVNS, and ABC, the EAPX outperforms in Solomon instances better than Cordeau instances. The Fig. 5 shows the results noted in Table 1 where the curve shows the obtainment results of the BKS and EAPX for Solomon instances.

As reported in Table 2, EAPX reaches the best-known solution only in two instances are pr06, pr10. It appears that EAPX under performs in Cordeau instances compared to ILS, GVNS, and ABC while the total average gap is 2.54%. Then, based on the obtained results, we can conclude that the performance of EAPX in solving TOPTW instances and the quality of the solution is proved. Our proposed approach is able to produce a good solution to the TOPTW.

Table 1. EAPX vs state-of-the art algorithms (Solomon instances)

Instance	m	BKS	EAPX		ILS		GVNS		ABC	
			Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)
c101	10	1810	1810	0.0	1720	5.0	1754	3.1	1786	1.3
c102	10	1810	1810	0.0	1790	1.1	1794	0.9	1810	0.0
c103	10	1810	1810	0.0	1810	0.0	1810	0.0	1810	0.0
c104	10	1810	1810	0.0	1810	0.0	1810	0.0	1810	0.0
c105	10	1810	1810	0.0	1770	2.2	1810	0.0	1782	1.5
c106	10	1810	1810	0.0	1750	3.3	1806	0.2	1786	1.3
c107	10	1810	1810	0.0	1790	1.1	1810	0.0	1784	1.4
c108	10	1810	1810	0.0	1810	0.0	1810	0.0	1792	1.0
c109	10	1810	1810	0.0	1810	0.0	1810	0.0	1810	0.0
r101	19	1458	1458	0.0	1441	1.2	1432.2	1.8	1457.4	0.0
r102	17	1458	1458	0.0	1450	0.5	1441.2	1.2	1455.4	0.2
r103	13	1458	1430	1.9	1450	0.5	1446.6	0.8	1455	0.2
r104	9	1458	1410	3.3	1402	3.8	1418.2	2.7	1437.8	1.4
r105	14	1458	1458	0.0	1435	1.6	1441.6	1.1	1458	0.0
r106	12	1458	1433	1.7	1411	1.2	1437.6	1.4	1458	0.0
r107	10	1458	1440	1.2	1431	1.9	1435	1.6	1452	0.4
r108	9	1458	1440	1.0	1430	1.9	1441.8	1.1	1451.8	0.4
r109	11	1458	1410	3.3	1432	1.8	1433.4	1.7	1449.4	0.6
r110	10	1458	1432	1.8	1419	2.7	1433.4	1.7	1449	0.6
r111	10	1458	1410	3.3	1410	3.3	1430.2	1.9	1449.8	0.6
r112	9	1458	1430	1.92	1418	2.7	1434.4	1.6	1448	0.7
rc101	14	1724	1632	5.3	1724	0.0	1690.2	2.0	1705.2	1.1
rc102	12	1724	1689	2.0	1718	0.3	1685	2.3	1705.2	1.1
rc103	11	1724	1686	2.2	1724	0.0	1709	0.9	1721	0.2
rc104	10	1724	1689	2.0	1724	0.0	1718	0.3	1723.4	0.2
rc105	13	1724	1640	4.9	1719	0.3	1689.8	2.0	1716	0.5
rc106	11	1724	1689	2.0	1716	0.5	1690.6	1.9	1706	1.0
rc107	11	1724	1724	0.0	1724	0.0	1718.4	0.3	1724	0.0
rc108	10	1724	1684	2.3	1719	0.3	1713	0.6	1720.6	0.2

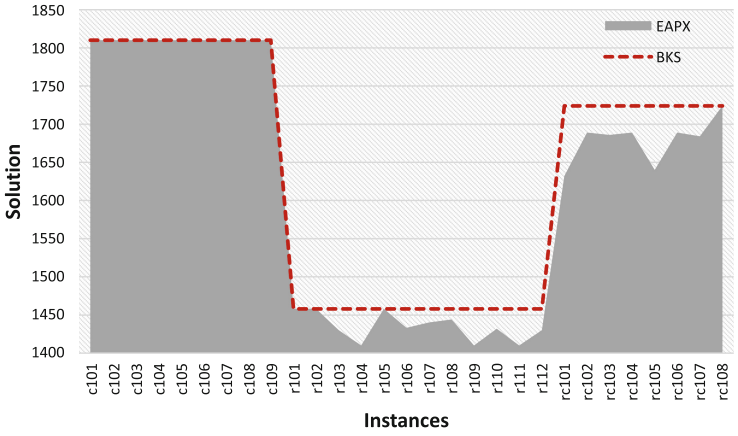

Fig. 5. Analysis of computational results for Solomon instances

Table 2. EAPX vs state-of the art algorithms (Cordeau instances)

Instance	m	BKS	EAPX		ILS		GVNS		ABC	
			Cost	Gap (%)	Cost	Gap (%)	Cost	Gap(%)	Cost	Gap (%)
pr01	3	657	603	8.2	608	7.5	608.4	7.4	617.4	6.0
pr02	6	1220	1192	2.3	1180	0.3	1198.8	1.7	1203.8	1.3
pr03	9	1788	1663	4.9	1738	2.8	1760.8	1.5	1764.6	1.3
pr04	12	2477	2400	3.1	2428	2.0	2467.4	0.4	2474.2	0.1
pr05	15	3351	3347	0.1	3297	1.6	3351	0.0	3351	0.0
pr06	18	3671	3671	0.0	3650	0.6	3670.6	0.0	3670.6	0.0
pr07	5	948	903	4.7	909	4.1	935	1.4	931.4	1.8
pr08	10	2006	1974	1.6	1984	1.1	2004.6	0.1	2006	0.0
pr09	15	2736	2724	0.4	2729	0.3	2736	0.0	2736	0.0
pr10	20	3850	3850	0.0	3850	0.0	3850	0.0	3850	0.0

5 Conclusion

In this paper, we have proposed a new solution for the team orienteering problem with time windows using the EAPX method. This algorithm employs the generation of the initial population, the local search operator, and the partition crossover method. Besides, we have investigated the performance of the EAPX using a comparative study with state of the art methods. The experimental results have shown the effectiveness of this meta-heuristic in solving TOPTW benchmark instances. The proposed approach performs, on good average compared to state-of-the-art algorithms and it reached the best known solutions.

References

Cura, T.: An artificial bee colony algorithm approach for the team orienteering problem with time windows. Comput. Ind. Eng. **74**, 270–290 (2014)

Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G.: A survey on algorithmic approaches for solving tourist trip design problems. J. Heuristics **20**(3), 291–328 (2014)

Labadie, N., Mansini, R., Melechovskỳ, J., Calvo, R.W.: The team orienteering problem with time windows: an LP-based granular variable neighborhood search. Eur. J. Oper. Res. **220**(1), 15–27 (2012)

Montemanni, R., Gambardella, L.M.: An ant colony system for team orienteering problems with time windows. Found. Comput. Decis. Sci. **34**(4), 287 (2009)

Ryan, D.M., Hjorring, C., Glover, F.: Extensions of the petal method for vehicle routeing. J. Oper. Res. Soc. **44**(3), 289–296 (1993)

Vansteenwegen, P., Souffriau, W., Berghe, G.V., Van Oudheusden, D.: Iterated local search for the team orienteering problem with time windows. Comput. Oper. Res. **36**(12), 3281–3290 (2009a)

Vansteenwege, P., Souffriau, W., Berghe, G.V., Van Oudheusden, D.: Metaheuristics for tourist trip planning. In: Sörensen, K., Sevaux, M., Habenicht, W., Geiger, M. (eds.) Metaheuristics in the Service Industry, vol. 624, pp. 15–31. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00939-6_2

Whitley, D., Hains, D., Howe, A.: Tunneling between optima: partition crossover for the traveling salesman problem. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, pp. 915–922. ACM (2009)