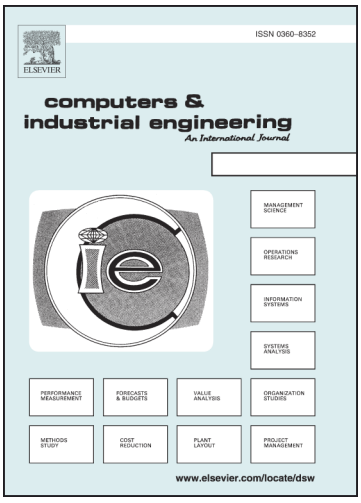

Team Orienteering Problem with Time Windows and Time-dependent Scores

Vincent F. Yu, Parida Jewpanya, Shih-Wei Lin, A.A.N. Perwira Redi

PII: S0360-8352(18)30584-9
DOI: https://doi.org/10.1016/j.cie.2018.11.044
Reference: CAIE 5534

To appear in: *Computers & Industrial Engineering*

Received Date: 27 December 2017
Revised Date: 26 August 2018
Accepted Date: 23 November 2018

Please cite this article as: Yu, V.F., Jewpanya, P., Lin, S-W., Perwira Redi, A.A.N., Team Orienteering Problem with Time Windows and Time-dependent Scores, *Computers & Industrial Engineering* (2018), doi: https://doi.org/10.1016/j.cie.2018.11.044

Team Orienteering Problem with Time Windows and Time-dependent Scores

Vincent F. Yu[a], Parida Jewpanya[a], Shih-Wei Lin[b,c,d*], A. A. N. Perwira Redi[a]

[a]*Department of Industrial Management, National Taiwan University of Science and Technology, Taiwan*

[b]*Department of Information Management, Chang Gung University, Taiwan*

[c]*Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan, Taiwan*

[d]*Department of Industrial Engineering and Management, Ming Chi University of Technology, Taipei, Taiwan*

Abstract

This study investigates the team orienteering problem with time windows and time-dependent scores (TOPTW-TDS), which is a new variant of the well-known team orienteering problem with time windows (TOPTW). TOPTW-TDS builds a set of paths to maximize the collected scores. The score of visiting an attraction is different depending on the time of visit. A mathematical formulation is proposed for this new problem. Since the problem is NP-hard, a hybrid artificial bee colony (HABC) algorithm is proposed to find the optimal solutions for small instances and near-optimal solutions for larger instances. Three datasets (small, medium and large) are generated with 168 TOPTW-TDS instances. Computational results reveal that the proposed HABC algorithm can generate high-quality TOPTW-TDS solutions. The proposed HABC obtains the optimal solution of each small TOPTW-TDS instance. For medium and large TOPTW-TDS instances, HABC results are comparable with the SA results. Additionally, results from solving TOPTW benchmark instances indicate that the proposed HABC compares well with the state-of-the-art algorithms for TOPTW.

Keywords: Metaheuristics; Logistics; Team orienteering problem; Time window; Artificial bee colony; Time-dependent score.

* Corresponding Author. E-mail address: *swlin@mail.cgu.edu.tw* (S.-W. Lin)

1. Introduction

The team orienteering problem with time windows (TOPTW) and its extension have been widely studied for many years. In TOPTW, a set of locations is given, each with a score, a service time and a time window. The aim is to maximize the collected scores from visiting locations. Recently, there has been an increasing research interest in TOPTW because of its practicality and complexity (Deitch & Ladany, 2000; Garcia et al., 2013; Vansteenwegen et al., 2011a; Zhu et al., 2010). One famous application of TOPTW is the tourist trip design problem (TTDP), which helps tourists in assessing all possible attractive locations in a customized trip (Gavalas et al., 2015; Labadie et al., 2012; Rodríguez et al., 2012).

TTDP is a route-planning problem for tourists interested in visiting multiple points of interest (POIs). Typically, most tourists visit as many POIs as possible during their limited time (Vansteenwegen et al., 2009a). However, given the time limitations, they may not be able to visit all POIs. Thus, tourists generally will select those locations with high scores or valuable POIs (Vansteenwegen et al., 2011a). TTDP aims to help tourists determine the most valuable tours when visiting POIs in a customized trip.

When visiting a POI, the score collected is assumed to be the same regardless of the time the visit takes place. This means that visitors receive the same experience or value whenever they visit the POI. For instance, when visiting a beautiful beach, the tourist obtains the same experience in the morning, afternoon and night. Nevertheless, in reality, upon a visit to a POI, different experience or value is gained at different time. For example, visiting the beautiful beach on a sunny day at sunrise in early morning or at the sunset in late afternoon is in general more attractive than visiting the same beach at noon on the same day, and thus the former should be rewarded a higher score. In this case, the tourist trip design problem is further

complicated because tourists must decide when they should visit the POIs. Therefore, the present study extends TOPTW to propose the team orienteering problem with time windows and time-dependent scores (TOPTW-TDS) to tackle this new and practical problem. TOPTW-TDS maximizes the total collected scores based on the time-dependent scores in different time periods of the visited POIs.

The proposed TOPTW-TDS is illustrated with the following example. A tourist plans to visit a certain area with several POIs. At each POI, several time periods can be selected to visit. Each time period is associated with a recommendation factor that is used to calculate the score of the visited POI. TOPTW-TDS aims to maximize the total collected score under several constraint limitations, such as the time budget and time windows. The applications of TOPTW-TDS are not limited to the trip design problem. It can also be applied to humanitarian logistics, where volunteers selects the routes to visit villages or camps at the right time. The application of routing food trucks or sales representatives to serve customers is also one possible application of TOPTW-TDS.

TOPTW-TDS is a new and challenging problem. Since it is an extension of the TOPTW, and TOPTW is a non-polynomial-hard (NP-hard) problem, TOPTW-TDS is also NP-hard. Therefore, a hybrid artificial bee colony (HABC) algorithm is proposed to effectively solve TOPTW-TDS. The proposed HABC embedded the acceptance criterion of simulated annealing (SA) in a modified artificial bee colony (ABC) framework to reduce the chance of the search being trapped at local optima.

Many new variants of TOPTW have been proposed in the literauture. However, to the best of our knowledge, none of them considers the case that scores collected are influenced by the visiting time of the nodes. In summary, the contributions of this study are as follows.

- A new variant of the TOPTW, namely TOPTW-TDS, in which the score collected is associated with the time of visit is proposed.
- New benchmark instances for TOPTW-TDS are generated based on TOPTW benchmark instances.
- A hybrid artificial bee colony algorithm is proposed to effectively solve TOPTW-TDS.

The rest of this paper is organized as follows. Section 2 reviews literature related to the proposed problem and solution approach. Section 3 presents the problem definition, with a mathematical model and some numerical examples. Section 4 describes the proposed HABC for solving TOPTW-TDS. Section 5 provides experimental results. Finally, Section 6 summarizes the results of this study, and suggests some future research directions.

2. Literature review

The Orienteering Problem (OP), Team Orienteering Problem (TOP) and their extensions have been studied for many years (Schilde et al., 2009; Souffriau et al., 2008; Tsiligirides, 1984; Vansteenwegen & Van Oudheusden, 2007; Wang et al., 2008). OP comes from the game of Orienteering (Chao et al., 1996a), which is an outdoor sport typically played in a forested area. In this game, each player tries to visit the checking points to collect points or scores. OP can be extended to TOP (Chao et al., 1996b) in which the game is played in teams. Vansteenwegen et al. (2011b) provided an excellent review of OP, TOP, Orienteering Problem with Time Windows (OPTW) and TOPTW, while Tang and Miller-Hooks (2005) gave a detailed review on the published solution approaches for OP and TOP. A recent study on solution approaches for TOP (Nicola et al., 2018) proposed a more compact formulation so

that TOP can be solved by a branch-and-cut algorithm and more benchmark instances are solved to optimality.

In reality, the inclusion of time windows in OP and TOP are necessary for certain routing problems where each location should be visited within a predetermined time interval. For example, in TTDP, some attractions can only be visited within a specific time interval. Therefore, tourists must plan their visit to these attractions during their opening periods. Abbaspour and Samadzadegan (2011) introduced an extension of OPTW. The tour-planning problem is modeled as the time-dependent OPTW. The time-dependent tour planning in a large urban area considers multimodal transportations. This problem determines the sequences of visiting the attractions during a specific period via several transportation modes. To solve the problem, an evolutionary strategy based on genetic algorithm is proposed.

Many variants of orienteering problem were proposed based on the considerations of one or more conditions faced in practice. Garcia et al. (2013) introduced a personalized electronic tourist guide to help tourists decide which places to visit. They integrated public transportation into TTDP to model the time-dependent TOPTW and solved the new problem using the algorithm of Vansteenwegen et al. (2009b). Souffriau et al. (2013) proposed the multiconstraint TOP with multiple time windows. In this problem, each node is associated with additional knap-sack constraints, which limit the selection of nodes. In TTDP applications, these additional constraints correspond to budget limitations for entrance fees for each day (e.g., a maximum number of museums to visit in the first day). A hybrid solution method that integrates an iterated local search and a greedy randomized adaptive search procedure (GRASP) was proposed to solve this problem. In addition, a recent study by Yu et al. (2017) addressed the development of a tourist trip design application that considers multimodal transportation. The study presented a new variant of TOPTW, called the multi-

modal team orienteering problem with time windows (MM-TOPTW). Recent development on the orienteering problem can be found in the survey of Gunawan et al. (2016). This survey provided information on the latest variants of the OP, including the proposed solution approaches and the most recent applications of the OP.

Several algorithms have been applied to solve OP and its extensions. These include exact methods (Boussier et al., 2006; Fischetti et al., 1998) and heuristic methods. The basic orienteering problem, which is an NP-hard problem, can be solved by exact method, but it is very difficult and usually requires a very long computational time. Consequently, heuristic methods have been adopted to obtain a near optimal solution in a reasonable amount of time. Vansteenwegen et al. (2011b) describe in detail a solution method for OP and its extensions. Meta-heuristic approaches, such as tabu search (Tang & Miller-Hooks, 2005), Ant Colony Optimization (Ke et al., 2008; Montemanni & Gambardella, 2009), Variable Neighborhood Search (Campbell et al., 2011; Tricoire et al., 2010), Iterated Local Search Heuristic (Vansteenwegen et al., 2009c), Simulated Annealing (SA) (Lin & Yu, 2012) and hybrid approach (Labadie et al., 2010), deliver good results for some specified problems. Gunawan et al. (2017) showed that applying design of experiments to fine-tune algorithmic parameters can lead to further improvement of the quality of the solutions generated by metaheuristic approaches.

3. Mathematical Formulation

In TOPTW-TDS, there is a set N of nodes. Each node i is associated with a basic score S_i and several recommendation factors $f_{i,t}$ which vary depending on the time of visit. The starting point and the end point of each tour are fixed (node 1). The time needed to travel from node i

to j is given by $t_{i,j}$. Not all nodes can be visited due to the given time budget (T_{max}). The problem can be formulated as the following integer programming model.

Sets and Parameters

P	Set of paths
N	Set of nodes
T	Set of periods
S_i	Basic score of each node
O_i	Starting time of the time window of node i
C_i	Ending time of the time window of node i
b_t	Starting time of period t
e_t	Ending time of period t
$t_{i,j}$	Traveling time from node i to node j
$f_{i,t}$	Recommendation factor of visiting node i in time period t
v_i	Duration (Service time) of the visit at node i
M	A large constant
T_{max}	A given time budget

Decision Variables

$s_{i,p}$			The starting time of the service/visit at node i in path p
$x_{i,j,t}^{p}$	=	1	If, in path p, a visit to node i in time period t is followed by a visit to node j;
		0	Otherwise

$y_{i,t}^{p}$ = 1 If a node *i* is visited by a path *p* in time period *t*;

0 Otherwise

Objective Function

$$\text{Maximize} \sum_{p\in P}\sum_{i\in N\setminus\{1\}}\sum_{t\in T} S_i \cdot f_{i,t} \cdot y_{i,t}^{p} \quad (1)$$

Constraints

$$\sum_{j\in N\setminus\{1\}}\sum_{t\in T} x_{1,j,t}^{p} = \sum_{i\in N\setminus\{1\}}\sum_{t\in T} x_{i,1,t}^{p} = 1 \qquad \forall p \in P \quad (2)$$

$$\sum_{i\in N, i\neq k}\sum_{t\in T} x_{i,k,t}^{p} = \sum_{j\in N, k\neq j}\sum_{t\in T} x_{k,j,t}^{p} = \sum_{t\in T} y_{k,t}^{p} \qquad \forall k \in N\setminus\{1\}, p \in P \quad (3)$$

$$\sum_{p\in P}\sum_{t\in T} y_{i,t}^{p} \leq 1 \qquad \forall i \in N\setminus\{1\} \quad (4)$$

$$\sum_{i\in N\setminus\{1\}}\sum_{t\in T} v_i \cdot y_{i,t}^{p} + \sum_{i,j\in N, i\neq j}\sum_{t\in T} t_{i,j} \cdot x_{i,j,t}^{p} \leq T_{\max} \qquad \forall p \in P \quad (5)$$

$$s_{i,p} + t_{i,j} + v_i - s_{j,p} \leq M(1 - x_{i,j,t}^{p}) \qquad \forall i, j \in N, i \neq j, p \in P, t \in T \quad (6)$$

$$y_{i,t}^{p} \cdot O_i \leq s_{i,p} \qquad \forall i \in N, p \in P, t \in T \quad (7)$$

$$s_{i,p} \leq C_i \qquad \forall i \in N, p \in P \quad (8)$$

$$b_t \cdot y_{i,t}^{p} \leq s_{i,p} \qquad \forall i \in N, p \in P, t \in T \quad (9)$$

$$s_{i,p} \leq e_t \cdot (y_{i,t}^{p} + ((1 - y_{i,t}^{p}) \cdot M)) \qquad \forall i \in N, p \in P, t \in T \quad (10)$$

$$x_{i,j,t}^{p}, y_{i,t}^{p} \in \{0,1\} \qquad \forall i, j \in N, p \in P, t \in T \qquad (11)$$

The objective function (1) maximizes the total collected score. If node *i* is visited in time period *t* in path *p*, the score of the visit is calculated by multiplying the basic score of node *i* by the recommendation factor of visiting node *i* in time period *t*. Constraint (2) states that each path starts and ends in node 1. Constraint (3) ensures the connectivity of each path. Not all nodes are visited in Constraint (4) because of the limitation of T_{max} in Constraint (5). Constraint (5) restricts that the total travel time of each tour is limited by a constant T_{max}. Constraint (6) determines the time line of each path. It implies that if arc (*i*, *j*) is visited in period *t*, then the starting time of the visit at node *j* has to be greater than or equal to the time that the starting time of the visit at node *i* plus the travel time from node *i* to node *j* and the duration of the visit at node *i*. Constraints (7) and (8) restrict the start of the service to within the time window. Constraints (9) and (10) ensure that if node *i* is visited in time period *t*, then the starting time of the service is in $[b_t, e_t]$.

4. Solution Approach for TOPTW-TDS

Artificial Bee Colony is one of the newly developed popular metaheuristic algorithms. After its invention by Karaboga (2005), ABC has been successfully applied to many real-world problems in a wide variety of fields such as scheduling (Bulut & Tasgetiren, 2014; Yurtkuran & Emel, 2016), data mining (Herawan et al., 2017; Wan et al., 2017), vehicle routing (Ng et al., 2017; Zhang et al., 2017), and control engineering (Sahed et al., 2016). For more details regarding this algorithm, readers are referred to an excellent survey of ABC and its applications (Karaboga et al., 2014).

ABC is an optimization algorithm inspired by the intelligent foraging behavior of a honey bee swarm. The bees in the algorithm are grouped into three types: employed bees,

onlookers and scouts. A bee currently searching for food is called an employed bee. An onlooker waits on the dance area to choose a food source; while a scout randomly search for a new food source. An employed bee becomes a scout after abandoning a food source. The exploration process of the search is controlled by the scouts; while the exploitation process is performed by employed bees and onlookers.

In ABC, each solution is regarded as a food source whose fitness is the amount of nectar in the food resource. The numbers of employed bees and onlookers are usually the same as the number of food sources. In other words, each food source only employs one bee. ABC is an iterative procedure similar to many population-based natural-inspired algorithms. The algorithm usually starts with a population of randomly generated solutions (food sources). A generation of ABC proceeds as follows. One employed bee is placed on each food source. Depending on the nectar amount carried by each food source, an appropriate number of onlookers are placed there. Scouts are sent to the search area to explore new food sources. In the end, the best food source found so far is stored in memory. This process is repeated until the termination condition is satisfied.

This study hybridizes the ABC algorithm with the SA-based acceptance rule to ensure that the search procedure can effectively escape from local optima. The following subsections describe in detail the components of the proposed HABC algorithm for TOPTW-TDS. First, the solution representation (encoding scheme) and the procedure for generating initial solutions are presented. Then the fitness value, neighborhood structure, and functions of employed bees and onlookers are explained. Finally, the complete procedure of the proposed HABC heuristic is described.

4.1 Representation of Solution and Initial Solutions

A solution is represented by a string of numbers consisting of a permutation of n locations denoted by the set $\{1,2,\ldots,n\}$ and $r-1$ zeros, followed by the n time period values representing the starting times of the services at individual locations, where r denotes the number of routes. The $r-1$ zeros are employed to separate routes. Therefore, the total number of elements in a solution is $2n + r - 1$. The i^{th} non-zero number in the first $n + r - 1$ positions denotes the i^{th} location to be serviced, and the last n elements determine the service time period for each of the n locations. Thus, the first non-zero entry in the solution representation indicates the first location of the first tour. Then, from left to right, other locations are appended to the tour one at a time, provided that the time window constraint at depot or any location is not violated.

If adding a location to the tour violates time window constraint of the location or depot, then the tour skips this location and considers the next location. If a customer arrives before the location's service time window, then the service is delayed until the location's service time window starts. The zeros in the solution representation terminate current tour and start a new tour whenever feasible.

For instance, the solution representation in Fig. 1 can be decoded as follows. Ten locations are serviced by two tours. The visiting sequence of route 1 is 5-8-2-6-4-7, and the visiting sequence of route 2 is 1-9-3-10. The service time periods are 1, 2, 1, 3, 1, 2, 3, 1, 2, and 3 for locations 1 to 10, respectively. In this case, visiting location 6 in route 1 and location 3 in route 2 will violate the time window constraint, and thus they cannot be served. Therefore, the visiting sequence of route 1 and route 2 is 5-8-2-4-7 and 1-9-10, respectively.

[Insert Figure 1 about here]

The initial solutions are randomly generated.

4.2 Fitness Value

Fitness value indicates the quality of a solution. The fitness value of a solution π is defined as $fit(\pi) = TS(\pi)$, where $TS(\pi)$ is the total score of π. The larger the fitness value, the higher the total score will be.

4.3 Neighborhood Solutions

The neighborhood solutions of a solution π is represented as *NS*(π), which is obtained by applying four types of moves to solution π, namely insertion, swap, inversion and change of service time period. The insertion is performed by randomly selecting a location of π and inserting it into the position immediately preceding another randomly selected location of π. The swap is performed by randomly selecting two locations of π, and then swapping their positions. The inversion is performed by randomly choosing a substring of π, and then reversing the visiting sequence of locations in the chosen substring. The service time period is changed by randomly choosing a location with at least two available service time periods, and then randomly changing its current service time period to another acceptable service time period.

The probability of choosing each move is set to 0.25. Note that because of the solution presentation scheme, the solutions resulted from the moves are always feasible.

4.4 *Employed Bee*

If the solution π^i corresponding to an employed bee x_i equals π_{best}, then a new solution π^i_{new} is generated from the neighborhood of π^i as described in Section 4.3.

An SA-based acceptance rule is adopted to guide the search process to escape from local optima. It is performed as follows. If the newly generated neighborhood solution π_{new}^{i} has a fitness value no worse than π^{i}, then it replaces π^{i} and becomes a new food source of the population. If the fitness value of π_{new}^{i} is worse than that of π^{i}, then π_{new}^{i} is accepted with a small probability. Let Δ denote the difference between the fitness values of π_{new}^{i} and of π^{i} (Δ = (TS(π_{new}^{i})−TS(π^{i}))/TS(π^{i})). The probability of replacing π^{i} with π_{new}^{i} is $e^{\Delta/T}$, where T is the current temperature. This is performed by randomly generating $r \in [0,1]$, and replacing π^{i} by π_{new}^{i} if $r < e^{\Delta/T}$. Because this algorithm adopts the SA-based acceptance rule, it does not use the scout phase.

4.5 Onlooker

An onlooker evaluates nectar information gathered from all the employed bees, and then selects a solution (food source) π^{k} with a probability $p_k = f_k / \sum_{k=1}^{SN} f_k$, where f_k is the nectar amount (fitness value) of π^{k}, and *SN* is the number of food sources. A higher f_k gives a larger probability of selecting π^{k}. After a solution π^{k} is chosen, a new solution π_{new}^{k} is generated based on the procedure described in Section 4.3. If π_{new}^{k} is better than π^{k}, then π_{new}^{k} will replace π^{k}. Otherwise, the SA-based acceptance rule is applied to π^{k}.

4.6 Procedure of the Proposed HABC Heuristic

The proposed HABC heuristic uses six parameters: *SN*, $G_{\max}$, $G_{\text{non-improving}}$, T_0, N_{iter} and α. *SN* is the number of food sources, which is the same as the total number of employed bees and onlookers. Therefore, the size of the colony is $2 \times SN$. G_{max} is the maximum number of

generations. $G_{\text{non-improving}}$ is the maximum number of consecutive generations during which the best objective function value found so far has not been improved. T_0 denotes the initial temperature of the SA procedure, N_{iter} is the number of generations performed at each temperature, and α represents the coefficient for the cooling schedule.

Figure 2 illustrates the pseudocode of the proposed HABC heuristic. The initial population $Pop = (\pi^1, \pi^2, ..., \pi^{SN})$ is generated randomly. The following steps in the main loop are performed at each iteration. First, the employed bees are placed on their food sources, and a new solution is generated from the neighborhood of food sources. If a new solution is better than or equal to the corresponding food source, then it replaces the original food source. Otherwise, an SA-based acceptance rule is used to determine whether to accept the worse solution. Second, the onlookers are placed on the food sources according to the amount of nectar of each source. A new solution is generated among the neighborhood of the chosen food source, and may replace the current food source according to its merit. Third, the best food source found so far (π^*) is recorded. Fourth, the current temperature T is decreased after running N_{iter} generations after the previous decrease, based on the formula $T \leftarrow \alpha T$, where $0 < \alpha < 1$. The termination condition is checked at the end of each iteration as follows. The algorithm stops when G is greater than G_{max} or when π^* has not been improved in $G_{non\text{-}improving}$ consecutive generations; otherwise, the procedure increases G by one and proceeds with a new generation. After the algorithm is terminated, π^* gives the (near-)global optimal solution.

[Insert Figure 2 about here]

5. Computational Study

The proposed HABC algorithm was coded in C, and tested on a computer equipped with an Intel Core 2 2.5 GHz CPU, which is comparable with the computational environment used by Montemanni and Gambardella (2009), Vansteenwegen *et al.* (2009b) and Lin and Yu (2012).

To verify its performance, the proposed HABC heuristic was compared with other existing state-of-the-art algorithms for TOPTW, namely the ACS based heuristic developed by Montemanni and Gambardella (2009) and the ILS-based heuristic proposed by Vansteenwegen *et al.* (2009b), the GRASP-ELS heuristic developed by Labadie et al. (2010), the GVNS based heuristic developed by Labadie et al. (2012), SA-based heuristic developed by Lin and Yu (2012), I3CS-based heuristic developed by Hu and Lim (2014) and ABC-based heuristic developed by Cura (2014). The proposed HABC heuristic was then applied to solve the TOPTW-TDS, and results were compared with those of CPLEX and the SA-based heuristic (Lin & Yu, 2012).

5.1 TOPTW and TOPTW-TDS Test Instances

The TOPTW test problems used in the computational experiment are taken from Righini and Salani (2009), Montemanni and Gambardella (2009), and Vansteenwegen *et al.* (2009b). These benchmark instances can be downloaded from http://www.mech.kuleuven.be/en/cib/op. Detailed descriptions of TOPTW test problems can be found in Lin and Yu (2012).

TOPTW-TDS instances were adapted from the well-known Solomon's benchmark instances of the vehicle routing problem with time windows (VRPTW). Each VRPTW instance has 25, 50 or 100 customers. The travel time between each pair of nodes equal the corresponding distance. The smaller instances with 25 and 50 customers are created by considering only the first 25 and 50 customers in the corresponding instance with 100

customers, respectively. The 56 VRPTW benchmark instances are grouped into six sets, labeled C1, C2, R1, R2, RC1 and RC2. The customer locations in these sets are randomly generated (R1 and R2), clustered (C1 and C2), or a mixture of randomly generated and clustered (RC1 and RC2). Instances in R1, C1, and RC1 have longer travel times, tighter time windows, and fewer customers per route than their counterpart in R2, C2, and RC2.

In the TOPTW-TDS dataset, the characteristics of the original instances are maintained by keeping the original coordinates of nodes, time windows, service times and scores (demands). Additionally, values of the new parameters, namely the time period and the recommendation factors, are added to each of the VRPTW instances to create TOPTW-TDS benchmark instances. The time periods can be regarded as different times of a day, such as morning, noon, afternoon and night. Each time period has a starting time, an ending time and a recommendation factor. The final score of a visit to a node in the objective function is calculated as the product of the node's basic score and the corresponding recommendation factor of the visit time period. The recommendation factors also depend on the time windows associated with the nodes. The recommendation factor for a time period outside the time window of a node is set to be zero; otherwise, the value is generated following the uniform random distribution $U(0,1)$.

5.2 Computational Results for TOPTW Instances

This subsection discusses the results of computational experiments designed for evaluating the performance of the proposed HABC algorithm. Since parameter setting may affect the performance of the algorithm, five instances were randomly selected from each of the four problem sets (m = 1, 2, 3, 4) for parameter analysis. Parameter combinations tested in the analysis are: T_0= {0.001, 0.002, 0.005, 0.01, 0.02, 0.03}; α={0.88, 0.90, 0.93, 0.95};

$G_{\max}$={80, 100, 120, 140}; $G_{non\text{-}improving}$={20, 30, 40, 50}; N_{iter}=(n+m−1)×B, where B={1000, 2000, 3000, 4000, 5000, 6000} and n and m are respectively the numbers of locations and tours. Preliminary testing results indicate that the best parameter values for HABC in solving TOPTW are T_0=0.02, N_{iter}=(n+m−1)×5000, α=0.90, $G_{\max}$=100, and $G_{non\text{-}mproving}$=40.

The proposed HABC heuristic was then tested on the TOPTW instances in the literature, and the results were compared with those obtained by ACO of Montemanni and Gambardella (2009), ILS of Vansteenwegen et al. (2009c), GRASP-ELS of Labadie et al. (2010), VNS of Tricoire et al. (2010), SSA of Lin and Yu (2012), GVNS of Labadie et al. (2012), I3CH of Hu and Lim (2014) and ABC of Cura (2014). The total scores obtained using HABC, ILS, GRASP-ELS, VNS, SSA, GVNS, I3CH and ABC for each benchmark instance were compared using the percentage gap (Gap) as follows.

$$Gap\ (\%) = \frac{TS_{BKS} - TS_h}{TS_{BKS}} \times 100\,\%,$$

where TS_h denotes, for each instance, the total score of the solution obtained using algorithm h, and TS_{BKS} denotes the corresponding total score of the best solution obtained by ILS, GRASP-ELS, VNS, SSA, GVNS, I3CH and ABC.

Table 1 summarizes comparative results, with the average (Ave.) and maximum results (Max.) of percentage gap (Gap %) and computational time (CPU) of each algorithm in the literature. The first column of the table shows the set of test instances of different number of tours (m=1, 2, 3, 4). The comparison indicates that HABC performed better than other algorithms when m=1 and m=2. For m = 3 and 4, HABC performed worse than I3CS. Additionally, HABC was tested on the instances for which the optimal solutions are known. When comparing HABC with ABC, it can be found that the largest Gap (%) obtained by

HABC is smaller than those of ABC and SSA. This indicates that by hybridizing SA-based acceptance rule with ABC algorithm, HABC allows the search procedure to escape effectively from local optima. Furthermore, HABC obtians 9 new best solutions. All the solutoins can be downloaded from http://swlin.cgu.edu.tw/data/HABC_TOPTW.xls. Table 2 compares overall performance of these algorithms on the two sets of instances with known optimal solutions. The result demonstrates that HABC achieved small average gaps, with 0.51% for the Solomon data set and 0.98% for the Cordeau data set. It can be concluded that HABC solves TOPTW effectively and efficiently.

[Insert Table 1 and 2 about here]

Test results indicate that I3CH (Hu & Lim, 2014) outperformed other approaches on solving TOPTW in terms of average solution quality. The I3CH framework comprises three components, namely local search, SA and route recombination, which is specifically designed for solving TOPTW. However, I3CH may not perform well on solving TOPTW-TDS, because it requires special treatments, especially in the solution representation.

5.3 Computational Results on TOPTW-TDS Instances

Two versions of HABC, FHABC and SHABC, are used to solve TOPTW-TDS. FHABC is the fast version of HABC, while SHABC is the slow version of HABC. Parameter analysis indicates that the best performing parameter combinations are: T_0=0.002, N_{iter}=(n+m−1)×1600, α=0.90, $G_{\max}$=100, $G_{non\text{-}improving}$=8 for FHABC, and T_0=0.002, N_{iter}=(n+m−1)×5000, α=0.98, $G_{\max}$=100, $G_{non\text{-}improving}$=40.

5.3.1 Comparing Results with CPLEX

The CPLEX solver was used to solve the mathematical model of TOPTW-TDS. The solver was terminated after 2 hours if it could not obtain an optimal solution. Table 3 presents the solution values and the computational times for instances with 25 nodes. CPLEX solver obtains optimal solutions to 82 out of 168 problems as shown in bold in Table 3. SHABC produces optimal solutions to 80 out of these 82 problems with known optimal solutions. For the other 86 problems without known optimal solutions, 36, 44, and 6 SHABC solutions are better than, equal, and worse than those of CPLEX, respectively. The average improving rate of SHABC is 0.451%, 0.344%, and 0.585% over CPLEX for m=1, 2, and 3, respectively. The overall average improving rate is 0.460%. Many factors may affect the computational time, for example: CPU speed, memory size, operation system, compiler, computer program, and precision. In general, our SHABC heuristic takes no more than 13 seconds to solve each of the small-scale problems, while CPLEX takes a much longer time to solve the same problems. In sum, SHABC obtains better solutions in less computation time compared to CPLEX.

[Insert Table 3 about here]

5.3.2 Comparing Results with Simulated Annealing

This section compares the performance of the proposed algorithm with simulated annealing, which has been successfully implemented to solve the TOPTW instances in Lin and Yu (2012). The TOPTW-TDS instances were solved using SA coded in the same computational environment as HABC. Two versions of SA were developed by Lin and Yu (2012). The version that uses a maximal computational time as the termination condition is called the fast simulated annealing (FSA), while the one using $N_{non\text{-}improving}$ (if the current best solution has not improved for $N_{non\text{-}improving}$ consecutive temperature decreases, the SA procedure is terminated) is called the slow simulated annealing (SSA). The parameter values

used for solving TOPTW-TDS are the same as those used by the SA in solving TOPTW, and the maximal computational time of FSA is the same as that used in FHABC.

Table 4 presents the comparison results between FHABC and FSA for the medium and large instances. The improving rate of FHABC over FSA is computed as $(TS_{HABC} - TS_{SA}) / TS_{SA} \times 100\%$, where TS_{HABC} and TS_{SA} is the solution values obtained by FHABC and FSA, respectively. As shown in Table 4, using the same computational time, the average improving rates (AIR) of FHABC over FSA for the medium instances are 0.300%, 0.896%, 0.416%, and 0.542% for *m*=1, 2, 3, and 4, respectively. Meanwhile, the AIRs of FHABC over FSA for the large instances are 0.394%, 0.537%, -0.017%, and 0.132% for *m*=1, 2, 3, and 4, respectively. The overall AIRs of FHABC over FSA are 0.539% and 0.262% for medium problems and large problems, respectively. Therefore, FHABC performs better than FSA in solving TOPTW.

[Insert Table 4 about here]

Table 5 compares the performance of SHABC with SSA for the medium and large instances. The AIRs of SHABC over SSA for medium instances are 0.022%, 0.37%, 0.198%, and 0.076% for *m*=1, 2, 3, and 4, respectively. Meanwhile, the AIRs of SHABC over SSA for large instances are -0.017%, 0.144%, 0.082%, and -0.013% for *m*=1, 2, 3, and 4, respectively. The overall AIRs of SHABC over SSA are 0.083% and 0.049% for medium problems and large problems, respectively. Therefore, the result shows that the SHABC performs better than SSA. SHABC and SSA use about the same computational time on average.

[Insert Table 5 about here]

Table 6 compares the performance of HABC with SA by the number of better solutions obtained. When comparing FHABC with FSA, the number of better solutions obtained by FHABC and FSA are 195 and 147, respectively. Meanwhile, the number of better solutions obtained by SHABC and SSA are 140 and 113, respectively. The number of better solutions obtained by HABC is greater than that of SA revealing that the proposed HABC algorithm performs better than SA in terms of solution quality.

[Insert Table 6 about here]

6 Conclusions and future research

This work studies the team orienteering problem with time windows and time-dependent scores, which is an extension of the well-known TOPTW. We propose a hybrid algorithm (HABC) to solve TOPTW-TDS by hybridizing ABC algorithm with an SA-based acceptance rule which improves the movement procedure of the basic ABC algorithm. The proposed algorithm is tested on TOPTW datasets from the literature. The results show that HABC compares well with other state-of-the-art algorithms. Additionally, three datasets with 168 TOPTW-TDS instances were generated. Both HABC and CPLEX can obtain the optimal solutions for small instances. For medium and large instances, HABC results are better than SA results.

TOPTW-TDS provides a challenging extension to TOPTW in practical applications. This extension can be implemented in other routing problems in which the demand varies over time. Additionally, the proposed algorithm is highly effective. The experiments indicate that the HABC algorithm is capable of producing high-quality TOPTW and TOPTW-TDS solutions. This approach may also be applied to other problems in a similar setting.

Furthermore, future research may also attempt to hybridize other algorithms to solve TOPTW-TDS.

References

Abbaspour, R. A., & Samadzadegan, F. (2011). Time-dependent personal tour planning and scheduling in metropolises. *Expert Systems with Applications, 38*(10), 12439-12452.

Boussier, S., Feillet, D., & Gendreau, M. (2006). An exact algorithm for team orienteering problems. *4or, 5*(3), 211-230.

Bulut, O., & Tasgetiren, M. F. (2014). An artificial bee colony algorithm for the economic lot scheduling problem. *International Journal of Production Research, 52*(4), 1150-1170.

Campbell, A. M., Gendreau, M., & Thomas, B. W. (2011). The orienteering problem with stochastic travel and service times. *Annals of Operations Research, 186*(1), 61-81.

Chao, I., Golden, B. L., & Wasil, E. A. (1996a). A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research, 88*(3), 475-489.

Chao, I., Golden, B. L., & Wasil, E. A. (1996b). The team orienteering problem. *European journal of operational research, 88*(3), 464-474.

Cura, T. (2014). An artificial bee colony algorithm approach for the team orienteering problem with time windows. *Computers & Industrial Engineering, 74*, 270-290.

Deitch, R., & Ladany, S. P. (2000). The one-period bus touring problem: Solved by an effective heuristic for the orienteering tour problem and improvement algorithm. *European journal of operational research, 127*(1), 69-77.

Fischetti, M., Gonzalez, J. J. S., & Toth, P. (1998). Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing, 10*(2), 133-148.

Garcia, A., Vansteenwegen, P., Arbelaitz, O., Souffriau, W., & Linaza, M. T. (2013). Integrating public transportation in personalised electronic tourist guides. *Computers & Operations Research, 40*(3), 758-774.

Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G., & Vathis, N. (2015). Heuristics for the time dependent team orienteering problem: Application to tourist route planning. *Computers and Operations Research, 62*, 36-50.

Gunawan, A., Lau, H. C., & Vansteenwegen, P. (2016). Orienteering Problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research, 255*(2), 315-332.

Gunawan, A., Lau, H. C., Vansteenwegen, P., & Lu, K. (2017). Well-tuned algorithms for the Team Orienteering Problem with Time Windows. *Journal of the Operational Research Society, 68*(8), 861-876.

Herawan, T., Hassim, Y. M. M., & Ghazali, R. (2017). Functional link neural network with modified artificial bee colony for data classification. [Article]. *International Journal of Intelligent Information Technologies, 13*(3), 1-14.

Hu, Q., & Lim, A. (2014). An iterative three-component heuristic for the team orienteering problem with time windows. *European Journal of Operational Research, 232*(2), 276-286.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2014). A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review, 42*(1), 21-57.

Ke, L., Archetti, C., & Feng, Z. (2008). Ants can solve the team orienteering problem. *Computers & Industrial Engineering, 54*(3), 648-665.

Labadie, N., Mansini, R., Melechovský, J., & Wolfler Calvo, R. (2012). The Team Orienteering Problem with Time Windows: An LP-based granular variable neighborhood search. *European Journal of Operational Research, 220*(1), 15-27.

Labadie, N., Melechovský, J., & Wolfler Calvo, R. (2010). Hybridized evolutionary local search algorithm for the team orienteering problem with time windows. *Journal of Heuristics, 17*(6), 729-753.

Lin, S.-W., & Yu, V. F. (2012). A simulated annealing heuristic for the team orienteering problem with time windows. *European Journal of Operational Research, 217*(1), 94-107.

Montemanni, R., & Gambardella, L. (2009). An ant colony system for team orienteering problems with time windows. *Foundations of computing and Decision Sciences, 34*, 287-306.

Ng, K. K. H., Lee, C. K. M., Zhang, S. Z., Wu, K., & Ho, W. (2017). A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion. *Computers and Industrial Engineering, 109*, 151-168.

Nicola, B., Renata, M., & Grazia, S. M. (2018). A branch-and-cut algorithm for the Team Orienteering Problem. *International Transactions in Operational Research, 25*(2), 627-635.

Righini, G., & Salani, M. (2009). Decremental state space relaxation strategies and initialization heuristics for solving the Orienteering Problem with Time Windows with dynamic programming. *Computers & Operations Research, 36*(4), 1191-1203.

Rodríguez, B., Molina, J., Pérez, F., & Caballero, R. (2012). Interactive design of personalised tourism routes. *Tourism Management, 33*(4), 926-940.

Sahed, O. A., Kara, K., Benyoucef, A., & Hadjili, M. L. (2016). An efficient artificial bee colony algorithm with application to nonlinear predictive control. *International Journal of General Systems, 45*(4), 393-417.

Schilde, M., Doerner, K. F., Hartl, R. F., & Kiechle, G. (2009). Metaheuristics for the bi-objective orienteering problem. *Swarm Intelligence, 3*(3), 179-201.

Souffriau, W., Vansteenwegen, P., Vanden Berghe, G., & Van Oudheusden, D. (2013). The multiconstraint team orienteering problem with multiple time windows. *Transportation Science, 47*(1), 53-63.

Souffriau, W., Vansteenwegen, P., Vertommen, J., Berghe, G. V., & Oudheusden, D. V. (2008). A personalized tourist trip design algorithm for mobile tourist guides. *Applied Artificial Intelligence, 22*(10), 964-985.

Tang, H., & Miller-Hooks, E. (2005). A TABU search heuristic for the team orienteering problem. *Computers & Operations Research, 32*(6), 1379-1407.

Tricoire, F., Romauch, M., Doerner, K. F., & Hartl, R. F. (2010). Heuristics for the multi-period orienteering problem with multiple time windows. *Computers & Operations Research, 37*(2), 351-367.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, 797-809.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Oudheusden, D. V. (2009a). A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research, 196*(1), 118-127.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Oudheusden, D. V. (2011a). The City Trip Planner: An expert system for tourists. *Expert Systems with Applications, 38*(6), 6540-6546.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Van Oudheusden, D. (2009b). Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research, 36*(12), 3281-3290.

Vansteenwegen, P., Souffriau, W., & Oudheusden, D. V. (2011b). The orienteering problem: A survey. *European Journal of Operational Research, 209*(1), 1-10.

Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., & Van Oudheusden, D. (2009c). Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research, 36*(12), 3281-3290.

Vansteenwegen, P., & Van Oudheusden, D. (2007). The mobile tourist guide: an OR opportunity. *OR Insight, 20*(3), 21-27.

Wan, S., Chang, S. H., Peng, C. T., & Chen, Y. K. (2017). A novel study of artificial bee colony with clustering technique on paddy rice image classification. *Arabian Journal of Geosciences, 10*(9), 215.

Wang, X., Golden, B. L., & Wasil, E. A. (2008). Using a genetic algorithm to solve the generalized orienteering problem *The vehicle routing problem: latest advances and new challenges*. (pp. 263-274): Springer.

Yu, V. F., Jewpanya, P., Ting, C.-J., & Redi, A. A. N. P. (2017). Two-level particle swarm optimization for the multi-modal team orienteering problem with time windows. *Applied Soft Computing, 61*, 1022-1040.

Yurtkuran, A., & Emel, E. (2016). A discrete artificial bee colony algorithm for single machine scheduling problems. *International Journal of Production Research, 54*(22), 6860-6878.

Zhang, D., Cai, S., Ye, F., Si, Y. W., & Nguyen, T. T. (2017). A hybrid algorithm for a vehicle routing problem with realistic constraints. *Information Sciences, 394-395*, 167-182.

Zhu, C., Hu, J. Q., Wang, F., Xu, Y., & Cao, R. (2010). On the tour planning problem. *Annals of Operations Research, 192*(1), 67-86.

5	8	**2**	6	**4**	**7**	0	1	9	3	10	1	2	3	4	5	6	7	8	9	10
Route 1						Route 2					Service time period									
											1	**2**	**1**	**3**	**1**	**2**	**3**	**1**	**2**	**3**

Figure 1 Example of TOPTW-TDS solution representation.

HABC (*SN*, G_{max}, $G_{non\text{-}improving}$, T_0, N_{iter} and α)
{
Generate initial population $Pop = (\pi^1, \pi^2, ..., \pi^{SN})$ randomly;
G=0; G_{non}=0; $T = T_0$;
Let π^* be the best solution;
Calculate the fitness of each food source $fit(\pi^i)$, $i \in Pop$;
Do ($G \le G_{\max}$ and $G_{non} \le G_{non-improving}$)
{
G=G+1; G_{non}= G_{non} +1;
//Place the employed bees on their food sources;
For $i = 1, 2, ..., SN$
{
Obtain new solution π^i_{new} by $NS(\pi^i)$;
If $\Delta = (TS(\pi^i_{new}) - TS(\pi^i))/TS(\pi^i) \ge 0$ { $\pi^i = \pi^i_{new}$;}
Else {
Generate r~U(0,1);
If $r < Exp(\Delta / T)$ { $\pi^i = \pi^i_{new}$;}
Else {Discard π^i_{new};}
}
}
//Place the onlooker bees on the food sources depending on their nectar amounts
For $i = 1, 2, ..., SN$
{
Selects a food source π^k depending on its probability value p_k calculated by $p_k = f_k / \sum_{k=1}^{SN} f_k$;
Obtain new solution π^k_{new} by $NS(\pi^k)$;
If $\Delta = (TS(\pi^i_{new}) - TS(\pi^i))/TS(\pi^i) \ge 0$ { $\pi^i = \pi^i_{new}$;}
Else {
Generate r~U(0,1);

```
            If r < Exp(Δ/T) { π^i = π^i_new ;}
            Else {Discard π^i_new ;}
          }
        }

        // Memorize the best food source found so far
        For i = 1, 2, ..., SN
          If fit(π^i_new) ≥ fit(π*) { π* = π^k ; G_non = 0; }

        //Temperature reduction
        If (G Mod N_iter=0) { T ← αT }
      }
    Output the best food source found so far ( π* );
}
```

Figure 2 Pseudo-code of the proposed HABC heuristic.

Table 1. Overall comparative results on TOPTW instances

Set		ACO		ILS		VNS		SSA		GRASP-ELS		GVNS		I3CS		ABC		HABC	
		Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)	Gap (%)	CPU (s)
m=1																			
Solomon 100	Ave	0.1	200.1	1.94	0.2	0.07	85.4	**0.05***	22.3	0.20	9.0	2.46	29.1	0.69	26.7	0.46	4.4	0.62	23.2
	Max	1.47	947.1	9.43	0.5	0.50	124.3	0.67	48.7	2.63	105.2	7.52	367.8	11.41	33.9	4.04	20.3	8.31	30.3
Solomon 200	Ave	2.03	1193.4	2.83	1.7	0.85	857.8	0.81	45.5	1.45	16.5	2.84	75.5	1.30	132.1	1.28	21.4	**0.54**	59.3
	Max	4.93	2641.7	5.80	2.8	2.62	1253.5	3.59	69.9	4.37	130.3	7.28	974.0	4.75	236.9	4.05	40.6	3.06	80.7
Cordeau 1-10	Ave	1.2	1626.6	4.72	1.8	1.08	822.1	0.97	107.6	1.44	5.0	1.61	12.4	1.05	37.4	1.34	78.5	**0.57**	122.3
	Max	4.31	2476.0	9.26	4.6	3.11	1772.4	2.69	269.5	3.33	9.1	4.98	32.7	4.38	117.4	3.30	247.4	2.35	280.5
Cordeau 11-20	Ave	11.8	887.7	9.49	2.0	3.30	1045.9	3.64	160.7	3.31	7.9	4.19	24.2	4.21	132.1	**2.93**	103.7	3.11	153.2
	Max	25.77	2312.7	17.06	5.3	7.08	2441.9	8.61	528.8	6.74	16.2	9.64	60.9	11.18	236.9	8.60	346.9	7.86	356.4
m = 2																			
Solomon 100	Ave	0.69	1278.6	1.91	0.9	1.03	68.8	**0.11**	35.2	1.35	26.6	1.70	73.9	0.45	69.3	0.39	13.0	0.24	38.2
	Max	2.63	2381.3	5.44	1.5	3.63	100.0	0.94	67.6	4.04	319.7	4.83	616.4	3.30	149.2	3.03	28.2	1.53	63.7
Solomon 200	Ave	3.14	2222.7	3.05	2.6	0.96	813.7	0.91	80.5	0.64	20.9	2.11	19.8	**0.42**	463.8	1.11	30.6	0.75	87.8
	Max	5.90	3057.4	5.72	5.1	2.07	1421.5	3.13	198.7	2.24	94.4	4.92	135.5	1.36	890.9	2.73	49.6	2.03	98.7
Cordeau 1-10	Ave	3.30	1889.7	5.96	4.8	3.62	524.8	2.18	171.3	1.93	19.5	1.51	39.1	0.84	247.1	2.54	149.7	**0.56**	210.8
	Max	7.61	3131.0	10.12	10.3	6.04	1018.9	5.11	417.7	4.60	35.5	3.18	94.3	2.71	447.0	6.00	445.7	1.67	497.6

Cordeau 11-20	Ave	5.65	2384.8	7.36	5.2	3.11	618.8	3.37	200.4	3.05	28.8	1.65	82.4	2.18	304.6	3.19	221.8	**1.60**	210.7
	Max	9.24	3240.7	13.65	11.0	4.98	1227.6	9.51	543.0	6.04	67.0	4.70	209.6	7.03	541.6	10.10	708.8	6.20	491.5
m = 3																			
Solomon 100	Ave	1.45	1421.7	2.40	1.5	1.22	68.9	0.44	48.4	1.60	35.0	1.87	91.1	**0.20**	135.9	0.63	22.5	0.48	50.6
	Max	3.37	2754.0	4.78	3.1	2.80	93.8	2.40	95.7	4.10	462.9	3.34	604.0	1.42	303.1	2.14	39.2	2.23	75.5
Solomon 200	Ave	0.84	1345.0	1.12	1.7	0.13	322.3	0.47	53.8	0.08	3.5	0.27	7.3	**0.01**	89.2	0.34	12.5	0.40	63.8
	Max	3.98	2564.3	4.30	3.4	1.30	814.4	3.31	124.2	0.92	22.6	1.57	21.0	0.29	981.2	2.87	45.3	3.31	103.6
Cordeau 1-10	Ave	3.98	2163.8	6.56	9.2	3.61	473.2	2.32	201.9	1.96	40.6	1.11	85.9	**0.34**	424.0	2.76	228.9	1.52	248.3
	Max	10.12	3187.7	10.27	27.9	6.65	923.5	6.10	598.0	4.30	86.6	2.19	198.0	0.81	731.1	7.30	658.4	3.40	575.8
Cordeau 11-20	Ave	6.58	2228.2	9.16	9.7	3.28	517.5	3.74	254.5	3.19	43.0	1.73	150.7	**1.08**	497.0	4.05	247.6	1.44	262.1
	Max	11.45	3094.8	12.63	28.2	5.56	1070.1	8.27	706.7	6.11	92.2	3.74	413.5	3.94	909.7	8.80	700.1	4.02	562.0
m = 4																			
Solomon 100	Ave	1.71	1523.0	3.16	2.4	1.61	66.8	0.53	61.5	1.58	39.9	1.89	86.6	**0.11**	199.5	2.71	183.1	0.42	69.0
	Max	4.77	2660.9	5.53	3.8	4.17	93.8	2.40	133.4	2.91	450.8	3.56	513.0	0.71	373.1	6.60	826.3	1.59	88.3
Solomon 200	Ave	0.02	245.4	**0.00**	1.0	**0.00**	141.2	**0.00**	0.1	**0.00**	0.0	**0.00**	0.5	**0.00**	0.2	**0.00**	0.5	**0.00**	31.7
	Max	0.55	2327.4	0.00	2.1	0.00	294.6	0.00	0.7	0.00	0.2	0.06	3.3	0.00	0.2	0.00	2.0	0.00	44.4
Cordeau 1-10	Ave	3.36	2447.7	6.86	14.1	3.34	403.2	1.99	255.7	2.29	45.8	1.40	127.3	**0.13**	566.5	3.29	233.3	1.29	291.7
	Max	7.09	3371.9	9.41	38.7	7.13	800.3	4.63	709.6	3.94	94.6	3.74	309.9	1.08	1044.4	5.50	725.7	3.55	637.5
Cordeau 11-20	Ave	6.02	2583.5	8.14	13.7	3.15	408.0	3.62	282.9	3.08	65.3	2.46	232.6	**0.11**	728.6	3.66	267.6	1.73	289.1
	Max	11.73	3293.9	12.15	35.9	6.34	862.4	7.81	755.4	7.14	149.6	6.37	632.0	1.06	1196.6	6.60	826.3	3.44	634.0

* Bold numbers indicate that the minimum gap is attained by the corresponding algorithm.

Table 2. Overall comparison on TOPTW instances with known optimal solutions

Set		ILS		VNS		SSA		GRASP-ELS		GVNS		I3CS		ABC		HABC	
		Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)
Solomon 100	Ave	1.80	3.2	0.45	20.9	0.59	79.1	0.55	65.7	1.14	29.7	0.03	393.8	0.53	41.24	0.51	106.21
	Max	4.97	5.6	1.69	28.7	2.76	175.3	1.87	178.6	3.09	58.7	0.27	3488.4	1.55	64.90	1.80	146.58
Cordeau 1-10	Ave	2.32	30.4	1.30	71.7	1.04	473.9	0.92	71.5	1.25	51.3	0.78	326.7	1.05	181.13	0.98	413.71
	Max	7.46	106.1	6.71	86.0	6.54	1526.5	5.78	341.2	7.40	196.4	5.78	1383.7	6.03	752.00	5.78	944.45

Table 3. Computational results for small TOPTW-TDS instances (25 nodes)

Instance Name	Objective Value						Computational time (s)					
	P=1		*P* = 2		*P* = 3		*P* = 1		*P* = 2		*P* = 3	
	CPLEX	SHABC	CPLEX	SHABC	CPLEX	SHABC	CPLEX	SHABC	CPLEX	SHABC	CPLEX	SHABC
c101_TDS	**250.0**	250.0	**400.0**	400.0	**460.0**	460.0	0.7	4.4	0.4	6.0	0.2	6.8
c102_TDS	288.0	288.0	**443.0**	443.0	**460.0**	460.0	7200.0	4.6	290.9	6.3	0.4	6.9
c103_TDS	268.0	272.0	417.0	420.0	447.0	447.0	7200.0	4.6	7200.0	7.0	7200.0	6.7
c104_TDS	275.0	276.0	400.0	406.0	435.0	435.0	7200.0	4.9	7200.0	6.3	7200.0	13.2
c105_TDS	**260.0**	260.0	**413.0**	413.0	**460.0**	460.0	1.1	4.4	7.5	5.9	0.9	6.7
c106_TDS	**240.0**	240.0	**403.0**	403.0	**460.0**	460.0	0.2	4.4	0.5	6.0	0.3	6.8
c107_TDS	**268.0**	268.0	**430.0**	430.0	**460.0**	460.0	3.9	4.4	40.5	5.8	0.6	6.5
c108_TDS	280.0	280.0	430.0	430.0	**459.0**	459.0	7200.0	4.4	7200.0	5.8	45.7	6.6
c109_TDS	274.0	274.0	416.0	416.0	428.0	428.0	7200.0	4.4	7200.0	5.8	7200.0	6.6
r101_TDS	**82.0**	82.0	**155.0**	155.0	**215.0**	215.0	0.7	3.5	0.7	4.5	0.7	5.5
r102_TDS	**120.7**	120.7	207.9	207.9	263.6	263.6	137.6	3.9	7200.0	5.9	7200.0	7.3
r103_TDS	128.6	128.6	206.3	213.7	259.7	271.6	7200.0	3.8	7200.0	5.4	7200.0	13.3
r104_TDS	112.0	112.0	201.1	201.1	251.7	261.8	7200.0	4.0	7200.0	5.6	7200.0	11.3
r105_TDS	**115.0**	115.0	**191.0**	191.0	**247.0**	247.0	0.8	3.7	2.4	4.8	6.6	6.0
r106_TDS	**127.7**	127.7	217.4	217.4	285.5	287.4	958.9	4.0	7200.0	5.7	7200.0	6.4
r107_TDS	129.1	129.1	212.6	218.8	273.2	277.2	7200.0	4.1	7200.0	9.1	7200.0	9.6
r108_TDS	120.3	121.5	209.6	213.0	268.0	272.6	7200.0	3.9	7200.0	8.5	7200.0	13.9
r109_TDS	**110.8**	110.8	**207.9**	207.9	**272.6**	272.6	24.8	3.8	4.1	5.0	3.6	8.9
r110_TDS	134.6	134.6	217.8	217.8	278.3	278.3	7200.0	3.8	7200.0	7.3	7200.0	8.5
r111_TDS	117.9	117.9	213.6	216.8	278.4	278.2	7200.0	4.8	7200.0	6.4	7200.0	11.4
r112_TDS	126.9	130.8	214.6	223.5	262.1	284.2	7200.0	3.8	7200.0	5.3	7200.0	6.2
rc101_TDS	**170.0**	170.0	**331.0**	331.0	**479.0**	479.0	0.3	4.0	3.8	5.4	7.9	6.4
rc102_TDS	**183.0**	183.0	361.0	361.0	502.0	502.0	505.9	4.2	7200.0	5.4	7200.0	7.4
rc103_TDS	192.0	192.0	363.0	363.0	478.0	478.0	7200.0	4.1	7200.0	5.4	7200.0	8.5
rc104_TDS	208.0	208.0	362.0	366.0	439.0	477.0	7200.0	4.1	7200.0	5.3	7200.0	7.3
rc105_TDS	**187.0**	187.0	357.0	357.0	495.0	495.0	46.6	4.0	7200.0	5.4	7200.0	7.5
rc106_TDS	**177.0**	177.0	349.0	349.0	488.0	488.0	97.8	4.3	7200.0	5.5	7200.0	6.3
rc107_TDS	205.0	205.0	367.0	367.0	521.0	521.0	7200.0	3.8	7200.0	5.4	7200.0	8.8

rc108_TDS	190.0	192.0	353.0	362.0	463.0	479.0	7200.0	5.2	7200.0	6.3	7200.0	8.7
c201_TDS	**450.0**	450.0	**460.0**	460.0	**460.0**	460.0	0.6	5.4	0.8	6.1	0.9	6.9
c202_TDS	**440.0**	440.0	**460.0**	460.0	**460.0**	460.0	1470.5	5.5	0.7	6.2	1.1	6.8
c203_TDS	416.0	416.0	456.0	456.0	**460.0**	460.0	7200.0	5.8	7200.0	8.9	1.0	7.2
c204_TDS	381.0	401.0	444.0	448.0	**460.0**	460.0	7200.0	5.6	7200.0	7.3	20.6	6.9
c205_TDS	**460.0**	460.0	**460.0**	460.0	**460.0**	460.0	1.9	5.5	0.9	6.2	0.2	6.9
c206_TDS	**460.0**	460.0	**460.0**	460.0	**460.0**	460.0	0.3	5.6	0.1	6.2	0.3	6.9
c207_TDS	**446.0**	446.0	**460.0**	460.0	**460.0**	460.0	24.5	5.7	1.0	6.0	0.3	6.7
c208_TDS	**455.0**	455.0	**460.0**	460.0	**460.0**	460.0	1.5	5.6	0.6	6.0	0.7	6.8
r201_TDS	**279.5**	279.5	**332.0**	332.0	**332.0**	332.0	11.8	5.4	5.9	6.3	1.4	6.8
r202_TDS	299.6	302.6	332.0	332.0	332.0	332.0	7200.0	9.1	7200.0	6.3	7200.0	6.8
r203_TDS	290.6	295.6	**332.0**	332.0	**332.0**	332.0	7200.0	7.0	6699.7	10.3	6.9	6.9
r204_TDS	292.7	291.7	331.3	332.0	332.0	332.0	7200.0	7.1	7200.0	12.3	7200.0	6.5
r205_TDS	297.1	297.1	**332.0**	332.0	**332.0**	332.0	7200.0	6.9	14.8	6.0	1.8	6.5
r206_TDS	302.5	310.0	**332.0**	332.0	**332.0**	332.0	7200.0	6.3	3.2	6.1	1.7	6.6
r207_TDS	306.7	307.8	**332.0**	332.0	**332.0**	332.0	7200.0	13.2	289.3	6.3	1.6	6.6
r208_TDS	313.5	309.2	330.9	330.9	**332.0**	332.0	7200.0	7.6	7200.0	6.3	22.0	6.4
r209_TDS	315.4	318.3	**330.8**	330.8	**332.0**	332.0	7200.0	6.3	18.5	6.5	1.0	6.4
r210_TDS	296.8	297.9	330.2	330.2	**332.0**	332.0	7200.0	5.4	7200.0	6.2	4.0	6.7
r211_TDS	290.8	291.2	314.1	314.1	321.8	322.6	7200.0	9.3	7200.0	7.6	7200.0	6.6
rc201_TDS	**428.0**	428.0	**534.0**	534.0	**540.0**	540.0	162.8	5.4	3.1	6.2	2.7	6.5
rc202_TDS	479.0	479.0	**530.0**	530.0	**540.0**	540.0	7200.0	5.5	48.7	6.2	0.5	7.1
rc203_TDS	467.0	467.0	530.0	528.0	**540.0**	540.0	7200.0	5.8	7200.0	8.4	1.0	8.2
rc204_TDS	459.0	491.0	**540.0**	540.0	**540.0**	540.0	7200.0	5.6	471.3	7.7	10.8	6.8
rc205_TDS	437.0	437.0	**518.0**	516.0	**530.0**	528.0	7200.0	5.5	1990.2	6.4	131.9	6.7
rc206_TDS	480.0	487.0	**540.0**	540.0	**540.0**	540.0	7200.0	5.6	0.8	6.0	0.4	6.4
rc207_TDS	500.0	505.0	**540.0**	540.0	**540.0**	540.0	7200.0	5.7	0.7	6.7	1.3	6.5
rc208_TDS	479.0	471.0	532.0	528.0	**540.0**	540.0	7200.0	5.6	7200.0	9.0	18.2	6.4
Ave time							4561.7	5.3	3905.4	6.5	2833.9	7.5
AIR (%)		0.451		0.344		0.585						

Table 4. Comparative results between FHABC and FSA for medium and large TOPTW-TDS instances

Instance name	Objective Value (n=50)								Objective Value (n=100)							
	P=1		P=2		P=3		P=4		P=1		P=2		P=3		P=4	
	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA
c101_TDS	270.0	270.0	490.0	480.0	650.0	650.0	760.0	750.0	860.0	860.0	1430.0	1430.0	1790.0	1793.0	1810.0	1810.0
c102_TDS	290.0	290.0	516.0	516.0	681.0	683.0	803.0	792.0	885.0	885.0	1411.0	1418.0	1717.0	1721.0	1781.0	1782.0
c103_TDS	285.0	281.0	504.0	499.0	655.0	662.0	766.0	765.0	857.0	850.0	1338.0	1322.0	1610.0	1618.0	1713.0	1705.0
c104_TDS	296.0	296.0	527.0	531.0	634.0	634.0	736.0	741.0	831.0	843.0	1264.0	1262.0	1547.0	1538.0	1644.0	1615.0
c105_TDS	300.0	300.0	520.0	520.0	680.0	680.0	770.0	760.0	890.0	890.0	1460.0	1450.0	1778.0	1800.0	1810.0	1810.0
c106_TDS	260.0	260.0	490.0	490.0	650.0	650.0	750.0	753.0	907.0	907.0	1450.0	1440.0	1776.0	1771.0	1809.0	1810.0
c107_TDS	310.0	310.0	530.0	530.0	690.0	690.0	780.0	780.0	901.0	901.0	1440.0	1437.0	1750.0	1753.0	1807.0	1807.0
c108_TDS	300.0	300.0	530.0	530.0	694.0	686.0	772.0	778.0	908.0	908.0	1441.0	1450.0	1765.0	1759.0	1810.0	1810.0
c109_TDS	281.0	278.0	485.0	480.0	632.0	631.0	738.0	738.0	745.0	750.6	1181.1	1183.5	1368.7	1371.2	1444.2	1445.5
r101_TDS	120.0	126.0	246.0	246.0	342.0	341.0	433.0	429.0	815.6	817.8	1259.0	1250.4	1413.4	1420.3	1454.8	1458.0
r102_TDS	163.6	163.6	311.6	307.5	415.8	421.8	517.6	511.1	808.4	811.9	1232.0	1239.8	1397.9	1384.2	1432.9	1430.7
r103_TDS	175.3	175.3	318.2	292.8	422.9	430.5	530.2	529.2	813.9	809.2	1203.8	1196.7	1344.2	1321.6	1394.5	1384.0
r104_TDS	167.5	166.0	299.3	287.7	401.5	423.6	498.5	506.9	763.0	765.0	1174.7	1172.2	1338.2			
r105_TDS																
r106_TDS																
r107_TDS																

r108_TDS

r109_TDS

r110_TDS

r111_TDS

r112_TDS

rc101_TDS

rc102_TDS

rc103_TDS

rc104_TDS

rc105_TDS

rc106_TDS

rc107_TDS

rc108_TDS

c201_TDS

c202_TDS

c203_TDS

c204_TDS

c205_TDS

c2
06
_
T
D
S

c2
07
_
T
D
S

c2
08
_
T
D
S

r2
01
_
T
D
S

r2
02
_
T
D
S

r2
03
_
T
D
S

r2
04
_
T
D
S

r2
05
_
T
D
S

r2
06
_
T
D
S

r2
07
_
T
D
S

r2
08
_
T
D
S

r2
09
_
T
D
S

r2
10
_
T
D
S

r2
11
_
T
D
S

rc
20
1_
T
D
S

rc
20
2_
T
D
S

rc
20
3_
T
D
S

rc
20
4_
T
D
S

rc205_TDS																
rc206_TDS																
rc207_TDS																
rc208_TDS																
Ave time	1.43	1.43	2.13	2.13	2.43	2.43	2.91	2.91	5.33	5.33	9.13	9.13	11.94	11.94	13.08	13.08
AIR (%)		0.300		0.896		0.416		0.542		0.394		0.537		-0.017		0.132

Table 5. Comparative results between SHABC and SSA for medium and large TOPTW-TDS instances

Instance name	**Objective Value (n=50)**								**Objective Value (n=100)**							
	P=1		P=2		P=3		P=4		P=1		P=2		P=3		P=4	
	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA	FHABC	FSA
c101_TDS	270.0	270.0	490.0	490.0	650.0	650.0	760.0	760.0	310.0	310.0	580.0	580.0	800.0	800.0	1000.0	1000.0
c102_TDS	290.0	290.0	516.0	516.0	691.0	691.0	811.0	811.0	338.0	338.0	628.0	628.0	888.0	888.0	1104.0	1114.0
c103_TDS	285.0	285.0	508.0	508.0	666.0	662.0	788.0	785.0	339.0	339.0	616.0	615.0	867.0	867.0	1096.0	1082.0
c104_TDS	296.0	296.0	531.0	531.0	648.0	653.0	749.0	740.0	341.0	341.0	622.0	623.0	866.0	878.0	1077.0	1082.0
c105_TDS	300.0	300.0	520.0	520.0	680.0	680.0	770.0	770.0	340.0	340.0	637.0	637.0	857.0	854.0	1053.0	1056.0
c106_TDS	260.0	260.0	490.0	490.0	650.0	650.0	753.0	753.0	330.0	330.0	610.0	610.0	850.0	850.0	1050.0	1050.0
c107_TDS	310.0	310.0	530.0	530.0	700.0	700.0	780.0	780.0	350.0	350.0	660.0	660.0	900.0	900.0	1100.0	1110.0
c108_TDS	300.0	300.0	530.0	530.0	694.0	686.0	784.0	784.0	350.0	350.0	660.0	660.0	890.0	890.0	1094.0	1090.0

c109_TDS	281.0	281.0	485.0	485.0	643.0	643.0	741.0	741.0	332.0	332.0	598.0	596.0	818.0	814.0	1012.0	1020.0
r101_TDS	126.0	126.0	246.0	246.0	342.0	342.0	433.0	433.0	197.0	197.0	340.0	340.0	471.0	471.0	597.0	596.0
r102_TDS	163.6	163.6	319.7	319.7	424.9	426.8	521.1	522.5	222.2	222.2	418.7	418.7	592.4	596.1	744.0	740.6
r103_TDS	175.3	175.3	322.9	322.9	436.5	438.3	538.4	539.1	229.0	229.0	422.8	422.8	601.1	599.1	749.8	760.5
r104_TDS	169.7	169.7	310.8	307.9	426.7	422.5	527.6	530.2	244.7	244.7	436.1	435.1	594.2	601.3	752.9	767.9
r105_TDS	149.0	149.0	271.5	271.5	382.3	382.3	476.3	476.3	228.2	228.2	423.0	423.0	575.0	578.2	722.3	720.1
r106_TDS	167.4	167.4	319.3	320.6	437.0	435.1	541.1	542.6	231.4	231.4	429.4	428.6	616.5	616.9	791.4	793.1
r107_TDS	182.0	182.0	339.2	339.2	465.2	463.8	553.2	550.7	246.4	246.4	445.4	447.8	629.0	641.5	792.6	809.3
r108_TDS	168.1	168.1	316.5	316.8	448.3	453.1	555.2	563.0	239.7	242.9	438.0	441.2	606.9	610.0	767.7	776.6
r109_TDS	161.0	161.0	310.3	310.3	424.4	424.4	525.0	525.0	224.4	224.4	416.4	416.4	589.6	589.6	745.6	738.2
r110_TDS	174.7	174.7	324.7	324.7	441.5	441.5	520.1	522.8	210.9	210.9	404.2	404.2	572.4	568.1	728.2	727.4
r111_TDS	173.0	173.0	328.0	328.0	448.5	448.5	544.8	547.8	243.3	243.3	448.0	446.6	626.5	627.1	780.4	782.6
r112_TDS	181.0	180.7	330.0	330.0	438.6	439.2	527.3	531.0	246.4	246.4	435.5	428.0	611.0	613.1	767.2	759.3
rc101_TDS	176.0	176.0	347.0	347.0	517.0	517.0	680.0	680.0	209.0	209.0	401.0	401.0	588.0	584.0	755.0	759.0
rc102_TDS	202.0	202.0	390.0	390.0	572.0	571.0	723.0	723.0	238.5	238.5	463.5	463.5	653.6	648.9	841.0	841.0
rc103_TDS	227.0	227.0	417.0	417.0	580.0	579.0	724.0	726.0	237.2	237.2	467.2	467.2	664.5	662.0	846.9	847.1
rc104_TDS	212.0	212.0	408.0	410.0	565.0	553.0	710.0	685.0	232.3	232.3	438.6	440.5	650.2	654.8	818.7	816.2
rc105_TDS	187.0	187.0	358.0	358.0	526.0	526.0	687.0	673.0	229.8	229.8	449.6	450.1	634.6	633.1	817.1	817.6
rc106_TDS	187.0	187.0	370.0	373.0	549.0	546.0	700.0	697.0	218.3	218.3	424.7	424.7	617.7	617.7	782.7	782.7
rc107_TDS	205.0	205.0	391.0	391.0	554.0	554.0	725.0	716.0	228.6	227.2	439.0	441.4	644.7	646.1	818.7	818.7
rc108_TDS	193.0	192.0	376.0	376.0	543.0	519.0	681.0	688.0	234.2	234.2	442.2	442.2	636.1	638.1	801.5	813.3
c201_TDS	650.0	650.0	860.0	860.0	860.0	860.0	860.0	860.0	860.0	860.0	1430.0	1430.0	1790.0	1793.0	1810.0	1810.0
c202_TDS	655.0	655.0	843.0	843.0	859.0	860.0	860.0	860.0	885.0	885.0	1411.0	1418.0	1717.0	1721.0	1781.0	1782.0
c203_TDS	621.0	618.0	798.0	801.0	840.0	835.0	855.0	855.0	857.0	850.0	1338.0	1322.0	1610.0	1618.0	1713.0	1705.0
c204_TDS	589.0	589.0	770.0	775.0	804.0	804.0	828.0	826.0	831.0	843.0	1264.0	1262.0	1547.0	1538.0	1644.0	1615.0
c205_TDS	670.0	670.0	860.0	860.0	860.0	860.0	860.0	860.0	890.0	890.0	1460.0	1450.0	1778.0	1800.0	1810.0	1810.0
c206_TDS	680.0	680.0	860.0	860.0	860.0	860.0	860.0	860.0	907.0	907.0	1450.0	1440.0	1776.0	1771.0	1809.0	1810.0

c2007_TDS

c208_TDSr201_TDSr202_TDSr203_TDSr204_TDSr205_TDSr206_TDSr207_TDS

r208_TDSr209_TDSr210_TDSr211_TDSrc201_TDSrc202_TDSrc203_TDS

rc204_TDSrc205_TDSrc206_TDSrc207_TDSrc208_TDS									774.3	748.4	1261.7	1259.8	1513.0	1504.7	1603.5	1573.8
Ave time	18.9	19.8	26.5	25.9	32.4	34.0	35.4	34.2	66.8	63.1	103.9	101.4	141.6	139.2	161.8	167.0
AIR (%)		0.022		0.037		0.198		0.076		-0.017		0.144		0.082		-0.013

Table 6. Summary of the comparative results between HABC and SA

		FHABC vs. FSA			**SHABC vs. SSA**		
Instances	*P*	FHABC Win	HABC=FSA	FSA Win	SHABC Win	SHABC=SSA	SSA Win
Medium	1	20	24	12	10	38	8
(50 nodes)	2	25	15	16	14	34	8

	3	22	17	17	21	25	10
	4	30	13	13	10	33	13
Large (100 nodes)	1	20	19	17	12	31	13
	2	26	11	19	23	16	17
	3	28	4	24	26	9	21
	4	24	3	29	24	9	23
Total		195	106	147	140	195	113

Research Highlights

- The team orienteering problem with time windows and time-dependent scores (TOPTW-TDS) is studied.
- A mathematical programming model for TOPTW-TDS is introduced.
- A hybrid artificial bee colony (HABC) algorithm is proposed for solving TOPTW-TDS.
- Computational study shows that the proposed HABC algorithm effectively solves TOPTW-TDS.