Available online at www.sciencedirect.com
ScienceDirect
Computers & Industrial Engineering 54 (2008) 648–665
computers & industrial engineering
www.elsevier.com/locate/dsw

Ants can solve the team orienteering problem

Liangjun Ke [a], Claudia Archetti [b], Zuren Feng [a,*]

[a] State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an, China
[b] Department of Quantitative Methods, University of Brescia, Brescia, Italy

Received 10 July 2007; received in revised form 1 October 2007; accepted 1 October 2007
Available online 6 October 2007

Abstract

The team orienteering problem (TOP) involves finding a set of paths from the starting point to the ending point such that the total collected reward received from visiting a subset of locations is maximized and the length of each path is restricted by a pre-specified limit. In this paper, an ant colony optimization (ACO) approach is proposed for the team orienteering problem. Four methods, i.e., the sequential, deterministic-concurrent and random-concurrent and simultaneous methods, are proposed to construct candidate solutions in the framework of ACO. We compare these methods according to the results obtained on well-known problems from the literature. Finally, we compare the algorithm with several existing algorithms. The results show that our algorithm is promising.

Keywords: Team orienteering problem; Ant colony optimization; Ant system; Heuristics

1. Introduction

In the team orienteering problem (TOP), a team of vehicles try to visit a set of points which are assigned a reward. Each vehicle starts from the starting point and finishes at the ending point within a prescribed time limit. Once a vehicle visits a point and is awarded the associated reward, no other vehicles can be awarded a reward for visiting the same point. The aim of the TOP is to maximize the total reward. The TOP was firstly studied by Butt and Cavalier (1994) under the name *multiple path maximum collection problem* and its current name was introduced by Chao, Golden, and Wasil (1996a). The TOP has been recognized as a model of many different real applications, such as the multi-vehicle version of the home fuel delivery problem (Golden, Levy, & Vohra, 1987), the recruiting of college football players (Butt & Cavalier, 1994), the sport game of team orienteering (Chao et al., 1996a), some applications of pickup or delivery services involving the use of common carriers and private fleets (e.g., Ballou & Chowdhury, 1980; Diaby & Ramesh, 1995; Hall & Racer, 1995) and the service scheduling of routing technicians (Tang & Miller-Hooks, 2005).

The TOP is a variant of the traveling salesman problem (TSP). It is closely related to other combinatorial optimization problems, such as vehicle routing problem (Tang & Miller-Hooks, 2005). Since the TOP is an

* Corresponding author.
E-mail addresses: Kelj163@163.com (L. Ke), archetti@eco.unibs.it (C. Archetti), fzr9910@xjtu.edu.cn (Z. Feng).

0360-8352/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2007.10.001

NP-hard problem (Chao et al., 1996a), the research efforts mainly focus on heuristics and metaheuristics. Butt and Cavalier (1994) presented a greedy procedure. Chao et al. (1996a) proposed a five-step method and a heuristic based on the stochastic algorithm of Tsiligirides (1984). A tabu search algorithm was proposed by Tang and Miller-Hooks (2005). Archetti, Hertz, and Speranza (2007) proposed two tabu search algorithms and a variable neighborhood search algorithm. Until now, only two exact algorithms have been proposed for the TOP: a column generation based algorithm (Butt & Ryan, 1999) and a branch and price based algorithm (Boussier, Feillet, & Gendreau, 2006). However, they can only solve problems of very limited size in a reasonable amount of time.

In this paper, an ant colony optimization (ACO) approach is proposed for the TOP. Although ACO has been successfully applied to a great variety of hard combinatorial optimization problems (Dorigo, Di Caro, & Gambardella, 1999; Dorigo & Stützle, 2004; Dorigo & Blum, 2005), this paper, as far as we know, proposes the first ACO-based algorithm for the TOP.

When solving a combinatorial optimization problem by ACO, a key point is to construct candidate solutions. In many problems, the construction procedures are natural. For example, in the TSP (Dorigo, Maniezzo, & Colorni, 1996), it is only necessary to choose one unselected city at each construction step. However, in the TOP, an ant must determine which vehicle to move and where to move. To deal with this problem, we proposes four methods, that is, the sequential, deterministic-concurrent and random-concurrent and simultaneous methods. A goal of this paper is to study these four methods in terms of computational time and solution quality. In addition, we compare the ACO-based algorithm with several existing algorithms.

The remainder of this paper is organized as follows: First, the TOP is introduced, and the background of ACO is presented. Then the proposed algorithm, which is called ACO–TOP, is presented in Section 3. Section 4 gives the experimental results. Finally, we conclude the main results.

2. Preliminaries

2.1. The formulation of the team orienteering problem

Given a complete graph $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the set of vertices, $E = \{(i,j) | i,j \in V\}$ is the set of edges. Each vertex i in V has a reward r_i. The starting point is vertex 1 and the ending point is vertex n, and $r_1 = r_n = 0$. For each edge (i,j) in E, a symmetric, nonnegative cost c_{ij} is associated with it, where c_{ij} is the distance between i and j. The corresponding TOP consists of finding m paths that start at vertex 1 and finish at vertex n such that the total reward of the visited vertices is maximized. Each vertex can be visited at most once. For each vehicle, the total time taken to visit the vertices cannot exceed a pre-specified limit $T_{\max}$. In the following, we will assume that there is a direct proportionality between the distance traveled by a vehicle and the time consumed by the vehicle. Thus, there is no actual difference if we consider $T_{\max}$ as a distance or a time. To avoid any confusion on this point, the value will be used throughout this paper as the maximum distance value.

The formulation for the TOP of which the starting and ending points are the same has been presented by Tang and Miller-Hooks (2005). It can be extended to the case where the starting and ending points may be different. Let $y_{ik} = 1$ $(i = 1, \ldots, n;\ k = 1, \ldots, m)$ if vertex i is visited by vehicle k, otherwise $y_{ik} = 0$. Let $x_{ijk} = 1$ $(i, j = 1, \ldots, n;\ k = 1, \ldots, m)$ if edge (i,j) is visited by vehicle k, otherwise $x_{ijk} = 0$. Since $c_{ij} = c_{ji}$, only $x_{ijk} (i < j)$ are defined. Let U be a subset of V. Then the TOP can be described as follows:

$$\max \sum_{i=2}^{n-1} \sum_{k=1}^{m} r_i y_{ik} \tag{1}$$

$$\text{subject to} \quad \sum_{j=2}^{n} \sum_{k=1}^{m} x_{1jk} = \sum_{i=1}^{n-1} \sum_{k=1}^{m} x_{ink} = m \tag{2}$$

$$\sum_{i<j} x_{ijk} + \sum_{i>j} x_{jik} = 2y_{jk} \quad (j = 2, \ldots, n-1;\ k = 1, \ldots, m) \tag{3}$$

$$\sum_{k=1}^{m} y_{ik} \leqslant 1 \quad (i = 2, \ldots, n-1) \tag{4}$$

$$\sum_{i=1}^{n-1}\sum_{j>i} c_{ij}x_{ijk} \leqslant T_{\max} \quad (k = 1, \ldots, m) \tag{5}$$

$$\sum_{\substack{i,j\in U\\ i<j}} x_{ijk} \leqslant |U| - 1 \quad (U \subset V \setminus \{1,n\};\ 2 \leqslant |U| \leqslant n-2;\ k = 1, \ldots, m) \tag{6}$$

$$x_{ijk} \in \{0,1\} \quad (1 \leqslant i < j \leqslant n;\ k = 1, \ldots, m) \tag{7}$$

$$y_{1k} = y_{nk} = 1, \quad y_{ik} \in \{0,1\} \quad (i = 2, \ldots, n-1;\ k = 1, \ldots, m) \tag{8}$$

where constraint (2) ensures that each vehicle starts at vertex 1 and ends at vertex n. Constraints (3) ensure the connectivity of each path. Constraints (4) ensure that each vertex (except vertex 1 and n) should be visited at most once, and constraints (5) describe the time restriction. Constraints (6) ensure that sub-paths are forbidden. Constraints (7) and (8) set the integral requirement on each variable.

2.2. Ant colony optimization

ACO is a class of population-based metaheuristics. It uses a colony of ants, which are guided by pheromone trails and heuristic information, to construct solutions iteratively for a problem. To solve a static combinatorial optimization problem with ACO, the main procedure can be described as follows: after all pheromone trails and parameters are initialized, ants construct solutions iteratively until a stopping criterion is reached. The main iterative procedure consists of two steps. In the first step, every ant constructs a solution according to the transition rule. Then a local search procedure can be adopted to improve one or more solutions. In the second step, the pheromone values are updated according to a pheromone updating rule.

Let us consider a maximization problem (S,f,Ω), where S is the set of candidate solutions, f is the objective function which assigns to each candidate solution an objective function (cost) value, and Ω is a set of constraints. The goal is to find a globally optimal feasible solution s^*, that is, a maximum cost feasible solution for the problem. In conclusion, a problem can be characterized by the following items (Blum & Dorigo, 2004)

(i) A finite set $C = \{c_1, c_2, \cdots, c_{|C|}\}$ of *solution components*.
(ii) A finite set X of *states* of the problem, defined in terms of possible sequences $x = \langle c_i, c_j, \ldots, c_h, \ldots \rangle$ of finite length over the elements of C. The length of a sequence, that is, the number of components in the sequence, is expressed by $|x|$. The maximum length of a sequence is bounded by a positive constant $n < +\infty$.
(iii) The set of candidate solutions S, with $S \subseteq X$.

To solve the problem, one should represent it by a completely connected and weighted graph, called construction graph, $G = (C, L, T)$, where the vertices are the components C, the set of edges L fully connects the components C, and T is a vector gathering pheromone trails. In ACO algorithms, ants deposit pheromone in order to attract other ants towards the corresponding area of the search space. Pheromone may be deposited on the edges or on the vertices. This paper mainly discusses the former case. As to the latter case, one can refer to the work by Blum and Dorigo (2004) for more details.

When constructing a solution, each ant is put on a starting point which is problem-dependent, and then wanders randomly from vertex to vertex in the graph. At each vertex, an ant probabilistically selects the next vertex on the basis of a decision policy or transition rule, which depends on the pheromone trails and the heuristic information on the edges. The widely used decision policy is

$$p(c_{k+1} = v|\tau, c_k = u) = \begin{cases} \frac{\tau(u,v)^\alpha \cdot \eta(u,v)^\beta}{\sum_{w\in C_u} \tau(u,w)^\alpha \cdot \eta(u,w)^\beta} & \text{if} \quad v \in C_u \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where c_k denotes the vertex selected at the kth constructing step, C_u is the set of vertices that can be selected from vertex u, $\tau(u,w)$ and $\eta(u,w)$ are the pheromone value and heuristic information on edge (u,w), respectively.

$\alpha(\alpha > 0)$ and β are two parameters which control the relative importance of pheromone trails and heuristic information.

In most applications, ants prevent from visiting infeasible vertices according to constraints Ω. However, penalty strategy may be occasionally used to deal with the constraints (Blum & Dorigo, 2004).

Once the ants have constructed their solutions, pheromone trails are updated. The first pheromone updating rule was introduced in ant system (AS) by Dorigo et al. (1996). More precisely, the rule is given as follows:

$$\tau(i,j)^{l+1} = \rho\tau(i,j)^{l} + \sum_{k=1}^{n_a} \Delta\tau_k(i,j) \tag{10}$$

where

$$\Delta\tau_k(i,j) = \begin{cases} F(s_k) & \text{if } (i,j) \text{ is visted by ant } k \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

$\tau(i,j)^l$ is the pheromone value of edge (i,j) at the lth cycle, ρ is a coefficient, $1-\rho$ is called evaporation rate $(0 \leqslant \rho < 1)$. Pheromone evaporation allows some past history to be forgotten, and helps diversify the search to new and hopefully more promising areas of the search space. s_k is the solution constructed by ant k, F is the quality function (Blum & Dorigo, 2004). n_a is the number of the ants. After pheromone trails are updated, those components which are visited by more ants and contained in higher-quality solutions will receive more pheromone and therefore will more likely be selected in future cycles.

Since AS was introduced, many improvements have been proposed to make ACO very effective. They differ from ant system mainly on the pheromone updating rule (Dorigo & Stützle, 2004; Dorigo & Blum, 2005). One of the most successful ACO variants is max–min ant system (MMAS) proposed by Stützle and Hoos (2000). The success of MMAS is mainly due to its sophisticated balance between the exploration and exploitation.

3. The proposed algorithm

ACO–TOP, the proposed algorithm for solving the TOP, is shown in Fig. 1. Although it follows the standard ACO algorithmic scheme for static combinatorial optimization problems, it has many new features which

Initialize the parameters, heuristic information η and the pheromone trails τ
$s_{ib} \leftarrow Null$
$s_{gb} \leftarrow Null$
$N_{ni} \leftarrow 0$
iteration $\leftarrow 0$
while *iteration* is less than N_C do
  for $i = 1$ to n_a do (n_a is the number of ants)
    $s_i \leftarrow$ **ConstructSolution**(τ, η) (see section 3.2)
    $s_i \leftarrow$ **LocalSearch**(s_i) (see section 3.4)
  end for
  $s_{ib} \leftarrow \arg\max(F(s_1), F(s_2), \cdots, F(s_{n_a}))$
  if $F(s_{ib}) > F(s_{gb})$
    $s_{gb} \leftarrow s_{ib}$
    $N_{ni} \leftarrow 0$
  else
    $N_{ni} \leftarrow N_{ni} + 1$
  end if
  PheromoneUpdate($\tau, s_{ib}, s_{gb}, N_{ni}$) (see section 3.3)
  iteration $\leftarrow$ *iteration* + 1
end while

Fig. 1. The ACO–TOP.

will be explained in this section. The algorithm performs as follows: at each cycle, each ant constructs a feasible solution and a local search procedure is applied to improve the solution. Subsequently, the pheromone trails are updated. The algorithm stops iterating when a maximum number of cycles N_C has been performed. In the following, we first discuss the definition of pheromone trails and heuristic information. Then we describe how to construct a feasible solution and the pheromone updating steps. Finally, a local search procedure is presented.

3.1. Pheromone trails and heuristic information

According to its definition, the TOP can be represented by a construction graph whose vertices are the vertices of the original problem. Each edge (i,j) is associate with an amount of pheromone $\tau(i,j)$ which represents the learned desirability of visiting a certain vertex j after another vertex i.

Suppose that the last vertex of the current (partial) path is i. For each vertex j that is not yet included in the path, the heuristic information $\eta(i,j)$ characterizes the desirability of edge (i,j). Since the objective of the TOP is to maximize the total reward within a prescribed time limit (or maximum distance value), those vertices which have higher rewards and which are closer to vertex i and n are more desirable. This observation motivates us to consider the three following attributes of vertex j: (1) the associated reward r_j, (2) the distance between i and j c_{ij}, (3) the degree of $\angle jin$, that is, $\arccos\theta_{ij}$ where $\theta_{ij} = (c_{ij}^2 + c_{in}^2 - c_{jn}^2)/2c_{ij}c_{in}$.

Based on these attributes, the heuristic information we used is given as follows:

$$\eta(i,j) = \frac{r_j}{c_{ij}} \exp(\gamma\theta_{ij}) \tag{12}$$

where γ is a parameter which determines the influence of θ_{ij}. When $\gamma = 0$, $\eta(i,j)$ is the measure given by Tsiligirides (1984).

Until now, many notable desirability measures have been developed (Chao, Golden, & Wasil, 1996b). Although our measure is simple, it can give very promising results.

3.2. Constructing a solution

In practice, it is only necessary to consider those candidate vertices which satisfy the following requirement (Chao et al., 1996a):

$$c_{i1} + c_{in} \leqslant T_{\max} \quad (2 \leqslant i \leqslant n-1) \tag{13}$$

This is because any path which contains one of other vertices will violate the time restriction. Without loss of generality, we assume that all vertices satisfy constraint (13).

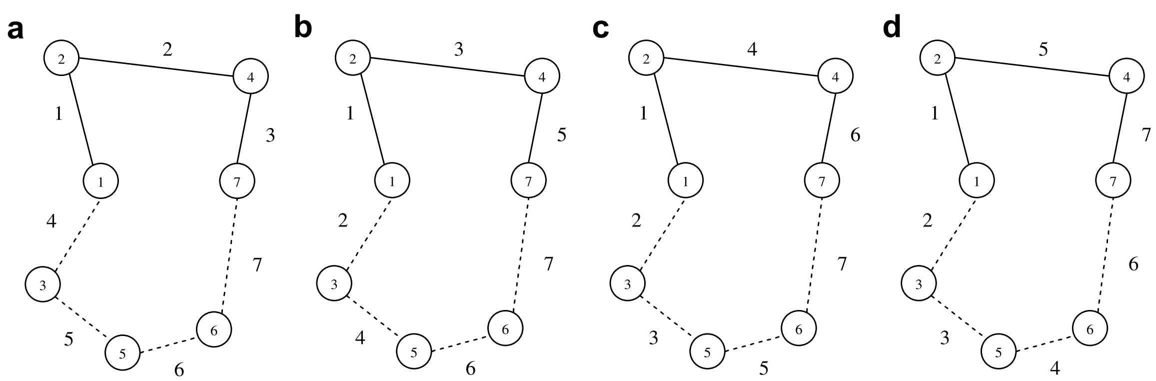

Fig. 2. An illustration of the four construction methods. (a) The sequential method. (b) The deterministic-concurrent method. (c) The random-concurrent method. (d) The simultaneous method. The solid lines show the paths of vehicle 1, and the dotted lines show the paths of vehicle 2. The numbers above the edges show the orders of construction steps.

Table 1
Results of the four methods

Group	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
1.2	**149.1**	**149.1**	5.2	**149.1**	**149.1**	4.9	**149.1**	**149.1**	**4.7**	**149.1**	**149.1**	4.9
1.3	**125.0**	**125.0**	5.8	**125.0**	**125.0**	5.2	**125.0**	**125.0**	**4.9**	**125.0**	**125.0**	5.2
1.4	**101.0**	**101.0**	6.4	**101.0**	**101.0**	5.7	**101.0**	**101.0**	**5.1**	**101.0**	**101.0**	5.7
2.2	**190.5**	**190.5**	2.9	**190.5**	**190.5**	2.8	**190.5**	**190.5**	**2.8**	**190.5**	**190.5**	2.8
2.3	**136.4**	**136.4**	3.1	**136.4**	**136.4**	3.0	**136.4**	**136.4**	**2.8**	**136.4**	**136.4**	2.9
2.4	**94.5**	**94.5**	3.4	**94.5**	**94.5**	3.2	**94.5**	**94.5**	**2.8**	**94.5**	**94.5**	3.2
3.2	**496.0**	**496.0**	6.1	**496.0**	495.8	5.8	**496.0**	495.8	**5.7**	**496.0**	**496.0**	5.8
3.3	**411.5**	**411.4**	6.5	**411.5**	411.2	6.0	**411.5**	411.1	**5.7**	**411.5**	411.2	6.0
3.4	**336.5**	**336.5**	6.9	**336.5**	**336.5**	6.3	**336.5**	336.2	**5.8**	**336.5**	336.3	6.3
4.2	**915.6**	**899.8**	34.1	908.4	898.3	30.6	909.5	897.3	**30.1**	911.8	899.6	30.8
4.3	**853.8**	841.1	37.2	847.7	841.1	31.6	848.4	840.1	**30.5**	848.2	**841.9**	31.8
4.4	**798.1**	783.0	40.5	795.9	784.1	33.4	791.4	780.4	**31.8**	795.2	**787.3**	33.8
5.2	**897.6**	**891.1**	15.6	896.4	890.5	14.0	896.2	892.2	**13.7**	896.2	890.7	14.1
5.3	**782.8**	**775.3**	17.3	780.4	774.6	15.0	781.2	774.5	**14.2**	781.0	774.2	15.0
5.4	**708.8**	**704.6**	19.3	707.7	698.5	16.2	706.3	698.2	**15.1**	705.6	698.2	16.3
6.2	**819.3**	815.4	14.8	818.7	814.1	13.3	818.7	814.8	**13.0**	**819.3**	**815.7**	13.4
6.3	**792.8**	786.0	16.7	790.5	785.6	14.5	790.5	784.7	**13.8**	791.3	**786.3**	14.6
6.4	**714.0**	701.3	18.2	**714.0**	699.1	15.2	**714.0**	699.5	**14.2**	**714.0**	**702.0**	15.3
7.2	**642.7**	637.4	26.0	641.5	**637.7**	22.2	641.0	637.1	**21.5**	641.2	637.4	22.1
7.3	**599.9**	**597.1**	30.8	599.4	595.5	24.8	598.6	594.6	**23.7**	599.2	596.0	25.0
7.4	**519.1**	**517.0**	34.5	518.2	516.3	26.8	518.4	516.2	**24.9**	518.4	516.3	26.9

The best results are indicated in bold.

During the construction of a solution for the TOP, an ant aims at choosing a feasible path for each vehicle. In detail, an ant iteratively chooses a vehicle and selects a feasible vertex for it, until all vehicles have reached the ending point (procedure **ConstructSolution** in Fig. 1). At each construction step, an ant can decide a vehicle and a point by means of one of the following methods:

Table 2
The rewards obtained by the seven algorithms

Group	ACO–TOP	CGW	TMH	GTP	GTF	FVF	SVF
1.2	**149.1**	148.5	148.8	**149.1**	**149.1**	**149.1**	**149.1**
1.3	125.0	**125.6**	124.7	125.0	125.0	125.0	125.0
1.4	**101.0**	99.3	**101.0**	**101.0**	**101.0**	**101.0**	**101.0**
2.2	**190.5**	190.0	190.0	**190.5**	**190.5**	**190.5**	**190.5**
2.3	**136.4**	135.9	135.9	**136.4**	**136.4**	**136.4**	**136.4**
2.4	**94.5**	**94.5**	**94.5**	**94.5**	**94.5**	**94.5**	**94.5**
3.2	**496.0**	488.5	492.0	494.5	**496.0**	**496.0**	**496.0**
3.3	**411.5**	403.0	408.0	**411.5**	**411.5**	**411.5**	**411.5**
3.4	**336.5**	332.5	335.0	**336.5**	**336.5**	**336.5**	**336.5**
4.2	915.6	875.7	895.1	904.9	908.5	914.0	**916.2**
4.3	853.8	815.1	844.3	845.5	852.5	853.0	**855.6**
4.4	798.1	766.1	784.6	800.1	802.3	801.7	**803.2**
5.2	**897.6**	890.6	886.8	892.6	897.4	895.8	897.0
5.3	783.4	776.6	775.8	781.4	**783.6**	**783.6**	**783.6**
5.4	**708.8**	696.0	699.0	707.5	**708.8**	**708.8**	**708.8**
6.2	**819.3**	814.9	818.2	813.8	818.7	**819.3**	**819.3**
6.3	**792.8**	787.5	783.0	**792.8**	**792.8**	**792.8**	**792.8**
6.4	714.0	**716.4**	712.8	714.0	714.0	714.0	714.0
7.2	**642.7**	633.9	633.5	639.6	641.4	640.6	642.5
7.3	**599.9**	585.5	592.5	596.7	597.7	597.1	599.3
7.4	**519.1**	497.4	514.6	517.2	516.9	516.9	518.9

The best results are indicated in bold.

Table 3
New best rewards obtained by ACO–TOP

Problem name	n	m	T_{max}	ACO–TOP	Previous best
p4.2.j	100	2	70.0	965	962
p4.2.p			100.0	1242	1241
p4.2.r			110.0	1288	1286
p4.2.s			115.0	1304	1301
p4.3.q		3	70.0	1252	1251
p4.3.t			80.0	1305	1304
p5.2.y	66	2	62.5	1645	1635
p7.2.i	102	2	90.0	580	579
p7.2.j			100.0	646	644
p7.3.l		3	80.0	684	683
p7.3.p			106.7	929	927
p7.3.t			133.3	1118	1117

Table 4
The maximal computational times of the seven algorithms

	ACO–TOP	CGW	TMH	GTP	GTF	FVF	SVF
Set 1	7.9	15.4	NA	10.0	5.0	1.0	22.0
Set 2	3.8	0.9	NA	0.0	0.0	0.0	1.0
Set 3	8.5	15.4	NA	10.0	9.0	1.0	19.0
Set 4	51.1	934.8	796.7	612.0	324.0	121.0	1118.0
Set 5	25.2	193.7	71.3	147.0	105.0	30.0	394.0
Set 6	20.3	150.1	45.7	96.0	48.0	20.0	310.0
Set 7	44.7	841.4	432.6	582.0	514.0	90.0	911.0

Table 5
new best rewards obtained by ACO–TOP when the final length of a path is rounded to one decimal place

Problem name	n	m	T_{max}	ACO–TOP
p1.2.e	32	2	12.5	50
p4.2.b	100	2	30.0	344
p4.2.k			75.0	1023
p4.3.c		3	23.3	194
p4.3.d			26.7	336
p5.3.f	66	3	10.0	135
p5.3.q			28.3	1080
p5.3.r			30.0	1145
p5.3.s			31.7	1225
p5.3.y			41.7	1600
p5.4.m		4	16.2	590
p5.4.p			20.0	780
p5.4.x			30.0	1500
p6.2.e	64	2	17.5	384
p6.2.j			30.0	972
p6.3.h		3	16.7	462
p6.4.k		4	16.2	558
p7.2.g	102	2	70.0	467
p7.4.q		4	85.0	912

(i) The *sequential* method, which does not replace the current vehicle until this vehicle has no feasible point to be visited. After a vehicle is determined, a point is chosen for it.
(ii) The *concurrent* method, which replaces the current vehicle by one of the other vehicles. After a vehicle is decided, a point is chosen for it. There are many ways to decide which vehicle have to replace the current vehicle. We consider two methods, called *deterministic-concurrent* and *random-concurrent* methods, which determinately and randomly schedule all vehicles respectively. In the former, the order of the vehicles is fixed, while the order is random in the latter.

(iii) The *simultaneous* method, which chooses one edge from those ones connecting the vehicles and the vertices. As a result, a vehicle and a point are simultaneously determined. The idea behind is to choose a promising edge at each step.

Table 6
Results for data set 1

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p1.2.b	15	15	2.8	15	15	2.4	15	15	2.1	15	15	2.5
p1.2.c	20	20	2.9	20	20	2.5	20	20	2.3	20	20	2.6
p1.2.d	30	30	3.2	30	30	2.8	30	30	2.5	30	30	2.8
p1.2.e	45	45	3.5	45	45	3.1	45	45	2.9	45	45	3.2
p1.2.f	80	80	4.1	80	80	3.8	80	80	3.5	80	80	3.8
p1.2.g	90	90	4.4	90	90	4.1	90	90	3.8	90	90	4.1
p1.2.h	110	110	4.8	110	110	4.4	110	110	4.2	110	110	4.5
p1.2.i	135	135	5.2	135	135	4.8	135	135	4.6	135	135	4.8
p1.2.j	155	155	5.4	155	155	5	155	155	4.8	155	155	5.1
p1.2.k	175	175	5.8	175	175	5.5	175	175	5.2	175	175	5.5
p1.2.l	195	195	6	195	195	5.7	195	195	5.5	195	195	5.7
p1.2.m	215	215	6.3	215	215	5.9	215	215	5.8	215	215	5.9
p1.2.n	235	235	6.5	235	235	6.2	235	235	6.1	235	235	6.2
p1.2.o	240	240	6.7	240	240	6.4	240	240	6.2	240	240	6.4
p1.2.p	250	250	6.8	250	250	6.5	250	250	6.3	250	250	6.5
p1.2.q	265	265	7	265	265	6.7	265	265	6.6	265	265	6.7
p1.2.r	280	280	7.2	280	280	6.9	280	280	6.8	280	280	7
p1.3.c	15	15	3.8	15	15	3.3	15	15	2.8	15	15	3.3
p1.3.d	15	15	3.8	15	15	3.3	15	15	2.8	15	15	3.3
p1.3.e	30	30	4.2	30	30	3.6	30	30	3.2	30	30	3.6
p1.3.f	40	40	4.4	40	40	3.8	40	40	3.4	40	40	3.8
p1.3.g	50	50	4.8	50	50	4.2	50	50	3.8	50	50	4.2
p1.3.h	70	70	5.1	70	70	4.5	70	70	4.2	70	70	4.5
p1.3.i	105	105	5.6	105	105	5	105	105	4.7	105	105	5
p1.3.j	115	115	5.8	115	115	5.2	115	115	4.9	115	115	5.2
p1.3.k	135	135	6.1	135	135	5.5	135	135	5.2	135	135	5.5
p1.3.l	155	155	6.4	155	155	5.8	155	155	5.6	155	155	5.8
p1.3.m	175	175	6.6	175	175	6	175	175	5.8	175	175	6
p1.3.n	190	190	6.9	190	190	6.3	190	190	6.1	190	190	6.3
p1.3.o	205	205	7	205	205	6.4	205	205	6.2	205	205	6.4
p1.3.p	220	220	7.1	220	220	6.5	220	220	6.3	220	220	6.6
p1.3.q	230	230	7.3	230	230	6.7	230	230	6.6	230	230	6.8
p1.3.r	250	250	7.5	250	250	6.9	250	250	6.7	250	250	6.9
p1.4.d	15	15	4.9	15	15	4.2	15	15	3.4	15	15	4.2
p1.4.e	15	15	4.9	15	15	4.2	15	15	3.4	15	15	4.2
p1.4.f	25	25	5.1	25	25	4.5	25	25	3.7	25	25	4.5
p1.4.g	35	35	5.4	35	35	4.6	35	35	4	35	35	4.6
p1.4.h	45	45	5.5	45	45	4.9	45	45	4.2	45	45	4.9
p1.4.i	60	60	6	60	60	5.2	60	60	4.7	60	60	5.2
p1.4.j	75	75	6.1	75	75	5.4	75	75	4.8	75	75	5.4
p1.4.k	100	100	6.6	100	100	5.8	100	100	5.3	100	100	5.8
p1.4.l	120	120	6.9	120	120	6.2	120	120	5.6	120	120	6.1
p1.4.m	130	130	6.9	130	130	6.2	130	130	5.7	130	130	6.2
p1.4.n	155	155	7.2	155	155	6.5	155	155	6	155	155	6.5
p1.4.o	165	165	7.4	165	165	6.6	165	165	6.2	165	165	6.6
p1.4.p	175	175	7.5	175	175	6.8	175	175	6.3	175	175	6.7
p1.4.q	190	190	7.6	190	190	6.9	190	190	6.5	190	190	6.9
p1.4.r	210	210	7.9	210	210	7.2	210	210	6.8	210	210	7.1

Table 7
Results for data set 2

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p2.2.a	90	90	2.3	90	90	2.1	90	90	2	90	90	2.1
p2.2.b	120	120	2.5	120	120	2.4	120	120	2.3	120	120	2.4
p2.2.c	140	140	2.6	140	140	2.6	140	140	2.5	140	140	2.5
p2.2.d	160	160	2.8	160	160	2.6	160	160	2.5	160	160	2.6
p2.2.e	190	190	2.9	190	190	2.8	190	190	2.7	190	190	2.8
p2.2.f	200	200	3	200	200	2.9	200	200	2.9	200	200	2.9
p2.2.g	200	200	3.1	200	200	3	200	200	3	200	200	3
p2.2.h	230	230	3.2	230	230	3.1	230	230	3	230	230	3.1
p2.2.i	230	230	3.2	230	230	3.2	230	230	3.1	230	230	3.2
p2.2.j	260	260	3.3	260	260	3.2	260	260	3.1	260	260	3.2
p2.2.k	275	275	3.4	275	275	3.3	275	275	3.2	275	275	3.3
p2.3.a	70	70	2.6	70	70	2.4	70	70	2.1	70	70	2.4
p2.3.b	70	70	2.7	70	70	2.6	70	70	2.3	70	70	2.5
p2.3.c	105	105	3	105	105	2.8	105	105	2.6	105	105	2.7
p2.3.d	105	105	3	105	105	2.8	105	105	2.6	105	105	2.8
p2.3.e	120	120	3.1	120	120	2.9	120	120	2.7	120	120	2.9
p2.3.f	120	120	3.1	120	120	2.9	120	120	2.8	120	120	2.9
p2.3.g	145	145	3.2	145	145	3	145	145	2.8	145	145	3
p2.3.h	165	165	3.3	165	165	3.1	165	165	3	165	165	3.1
p2.3.i	200	200	3.4	200	200	3.3	200	200	3.2	200	200	3.3
p2.3.j	200	200	3.5	200	200	3.4	200	200	3.2	200	200	3.3
p2.3.k	200	200	3.4	200	200	3.4	200	200	3.3	200	200	3.4
p2.4.a	10	10	3	10	10	2.7	10	10	2.2	10	10	2.7
p2.4.b	70	70	3.2	70	70	3	70	70	2.6	70	70	3
p2.4.c	70	70	3.2	70	70	3.1	70	70	2.6	70	70	3.1
p2.4.d	70	70	3.2	70	70	3.1	70	70	2.6	70	70	3.1
p2.4.e	70	70	3.3	70	70	3.1	70	70	2.6	70	70	3.1
p2.4.f	105	105	3.6	105	105	3.3	105	105	2.9	105	105	3.3
p2.4.g	105	105	3.5	105	105	3.3	105	105	2.9	105	105	3.3
p2.4.h	120	120	3.6	120	120	3.4	120	120	3	120	120	3.4
p2.4.i	120	120	3.6	120	120	3.4	120	120	3	120	120	3.4
p2.4.j	120	120	3.6	120	120	3.4	120	120	3	120	120	3.4
p2.4.k	180	180	3.8	180	180	3.6	180	180	3.3	180	180	3.6

In order to describe these four methods, we use the following notations.

u_i: the vertex where the ith vehicle settles at the kth construction step ($1 \leqslant i \leqslant m$);
C_{u_i}: the set of the feasible vertices which are unvisited by any vehicle and satisfy the following restriction:

$$\forall v \in C_{u_i}, L(t_i) + c_{u_i v} + c_{vn} \leqslant T_{\max} \tag{14}$$

where $L(t_i)$ is the length of the (partial) path t_i which is traveled by the ith vehicle. If the set C_{u_i} is empty, that is, there is no feasible point, then the ending point is chosen and the ith path is completed.

v_k: the vertex selected at the kth construction step;
q_k: the vehicle selected at the kth construction step.

In the sequential, deterministic-concurrent and random-concurrent methods, an ant probabilistically chooses a point according to the decision policy as follows:

$$p(v_{k+1} = v, q_{k+1} = j | C_{u_i}, 1 \leqslant i \leqslant m, q_k, \tau) = \begin{cases} \frac{\tau(u_j,v)^{\alpha} \cdot \eta(u_j,v)^{\beta}}{\sum_{w \in C_{u_j}} \tau(u_j,w)^{\alpha} \cdot \eta(u_j,w)^{\beta}} & \text{if} \quad v \in C_{u_j} \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Table 8
Results for data set 3

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p3.2.a	90	90	3.2	90	90	2.9	90	90	2.7	90	90	3
p3.2.b	150	150	3.8	150	150	3.5	150	150	3.2	150	150	3.5
p3.2.c	180	180	4.2	180	180	3.9	180	180	3.7	180	180	3.9
p3.2.d	220	220	4.6	220	220	4.3	220	220	4.1	220	220	4.3
p3.2.e	260	260	4.8	260	260	4.5	260	260	4.3	260	260	4.6
p3.2.f	300	300	5.2	300	300	4.8	300	300	4.6	300	300	4.8
p3.2.g	360	360	5.4	360	360	5.1	360	360	4.9	360	360	5.1
p3.2.h	410	410	5.7	410	410	5.4	410	410	5.2	410	410	5.4
p3.2.i	460	460	6	460	460	5.7	460	460	5.5	460	460	5.7
p3.2.j	510	510	6.3	510	510	5.9	510	510	5.8	510	510	6
p3.2.k	550	550	6.5	550	550	6.2	550	550	6.1	550	550	6.3
p3.2.l	590	590	6.7	590	590	6.4	590	590	6.3	590	590	6.4
p3.2.m	620	620	6.9	620	620	6.6	620	620	6.4	620	620	6.6
p3.2.n	660	660	7.1	660	660	6.8	660	660	6.6	660	660	6.8
p3.2.o	690	690	7.2	690	690	6.9	690	690	6.8	690	690	7
p3.2.p	720	720	7.4	720	720	7.1	720	720	7	720	720	7.2
p3.2.q	760	760	7.6	760	760	7.3	760	760	7.2	760	760	7.3
p3.2.r	790	790	7.7	790	790	7.5	790	790	7.4	790	790	7.5
p3.2.s	800	800	7.9	800	800	7.6	800	800	7.6	800	800	7.7
p3.2.t	800	800	8.1	800	800	7.8	800	800	7.7	800	800	7.8
p3.3.a	30	30	3.9	30	30	3.5	30	30	2.9	30	30	3.5
p3.3.b	90	90	4.3	90	90	3.8	90	90	3.3	90	90	3.8
p3.3.c	120	120	4.5	120	120	4.1	120	120	3.6	120	120	4.1
p3.3.d	170	170	5.1	170	170	4.5	170	170	4.1	170	170	4.5
p3.3.e	200	200	5.4	200	200	4.9	200	200	4.5	200	200	4.9
p3.3.f	230	230	5.6	230	230	5.1	230	230	4.8	230	230	5.1
p3.3.g	270	270	5.9	270	270	5.5	270	270	5.2	270	270	5.5
p3.3.h	300	300	6.2	300	300	5.6	300	300	5.4	300	300	5.7
p3.3.i	330	330	6.4	330	330	5.9	330	330	5.6	330	330	5.9
p3.3.j	380	380	6.6	380	380	6.1	380	380	5.8	380	380	6.1
p3.3.k	440	440	6.8	440	440	6.3	440	440	6.1	440	440	6.3
p3.3.l	480	480	7	480	480	6.5	480	480	6.3	480	480	6.5
p3.3.m	520	520	7.2	520	520	6.7	520	520	6.5	520	520	6.7
p3.3.n	570	570	7.4	570	570	6.9	570	570	6.7	570	570	6.9
p3.3.o	590	590	7.5	590	590	7.2	590	590	6.9	590	590	7.1
p3.3.p	640	640	7.7	640	640	7.5	640	640	7.1	640	640	7.3
p3.3.q	680	680	7.8	680	680	7.4	680	680	7.2	680	680	7.4
p3.3.r	710	710	8	710	710	7.5	710	710	7.4	710	710	7.6
p3.3.s	720	720	8	720	720	7.6	720	720	7.5	720	720	7.6
p3.3.t	760	760	8.1	760	760	7.7	760	760	7.6	760	760	7.7
p3.4.a	20	20	4.8	20	20	4.2	20	20	3.4	20	20	4.2
p3.4.b	30	30	4.9	30	30	4.3	30	30	3.6	30	30	4.4
p3.4.c	90	90	5.3	90	90	4.7	90	90	3.9	90	90	4.7
p3.4.d	100	100	5.4	100	100	4.8	100	100	4.1	100	100	4.8
p3.4.e	140	140	5.8	140	140	5.1	140	140	4.5	140	140	5.1
p3.4.f	190	190	6.2	190	190	5.5	190	190	4.9	190	190	5.5
p3.4.g	220	220	6.5	220	220	5.8	220	220	5.3	220	220	5.8
p3.4.h	240	240	6.7	240	240	5.9	240	240	5.5	240	240	6
p3.4.i	270	270	6.8	270	270	6.2	270	270	5.7	270	270	6.1
p3.4.j	310	310	7.1	310	310	6.4	310	310	6	310	310	6.4
p3.4.k	350	350	7.2	350	350	6.6	350	350	6.2	350	350	6.5
p3.4.l	380	380	7.4	380	380	6.7	380	380	6.3	380	380	6.7
p3.4.m	390	390	7.5	390	390	6.9	390	390	6.4	390	390	6.8
p3.4.n	440	440	7.7	440	440	7	440	440	6.7	440	440	7.1
p3.4.o	500	500	7.9	500	500	7.2	500	500	6.9	500	500	7.2
p3.4.p	560	560	8	560	560	7.3	560	560	7	560	560	7.3

(*continued on next page*)

Table 8 (*continued*)

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p3.4.q	560	560	8.1	560	560	7.6	560	560	7.2	560	560	7.5
p3.4.r	600	600	8.2	600	600	7.6	600	600	7.3	600	600	7.7
p3.4.s	670	670	8.4	670	670	7.7	670	670	7.5	670	670	7.8
p3.4.t	670	670	8.5	670	670	7.9	670	670	7.6	670	670	7.9

Note that vehicle q_k and q_{k+1} are the same in the sequential method, while they are different in the concurrent method.

In the simultaneous method, an ant randomly chooses a vehicle and a point by the following probability:

$$p(v_{k+1} = v, q_{k+1} = j | C_{u_i}, 1 \leqslant i \leqslant m, q_k, \tau) = \begin{cases} \frac{\tau(u_j,v)^\alpha \cdot \eta(u_j,v)^\beta}{\sum_{i=1}^{m}\sum_{w\in C_{u_i}} \tau(u_i,w)^\alpha \cdot \eta(u_i,w)^\beta} & \text{if} \quad v \in C_{u_j} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

Since only one vertex and one vehicle are selected in the simultaneous method, formula (16) can be computed relatively fast by calculating $\sum_{w\in C_{u_i}} \tau(u_i,w)^\alpha \cdot \eta(u_i,w)^\beta (1 \leqslant i \leqslant m)$. Suppose that vehicle j is chosen and vertex v is selected for it, for each vehicle $i (i \neq j)$, the set of its feasible vertices changes to $C_{u_i} \setminus \{v\}$.

In Fig. 2, we illustrate the four methods on a TOP with $n = 7$ and $m = 2$. The numbers on the edges represent the orders of construction steps. The sequential method is shown in Fig. 2a. From step 1 to step 3, three points are chosen for vehicle 1. Then four points are chosen for vehicle 2. As shown in Fig. 2b and c, two concurrent methods alternately choose a point for the two vehicles. Fig. 2d shows the simultaneous method where one vehicle is randomly chosen at each step.

3.3. The Pheromone updating

After each ant has constructed a solution, the pheromone trails are updated mainly according to MMAS (procedure **PheromoneUpdate** in Fig. 1). More formally, at the end of each cycle, the pheromone trail of each edge (u, v) is updated as follows:

$$\tau(u,v)^{l+1} = \rho\tau(u,v)^l + \Delta\tau(u,v) \tag{17}$$

$$\text{if} \quad \tau(u,v)^{l+1} < \tau_{\min}, \quad \text{then } \tau(u,v)^{l+1} = \tau_{\min} \tag{18}$$

$$\text{if} \quad \tau(u,v)^{l+1} > \tau_{\max}, \quad \text{then } \tau(u,v)^{l+1} = \tau_{\max} \tag{19}$$

where $\tau(u,v)^l$ is the pheromone value of edge (u, v) at the lth cycle. If (u, v) is visited by the best ant at the lth cycle, then $\Delta\tau(u,v)$ is equal to $F(s_{\text{best}})$, otherwise $\Delta\tau(u,v) = 0$. s_{best} may be the iteration-best solution s_{ib} or global-best solution s_{gb}. $F(x)$ is the quality function which is given as follows:

$$F(x) = \sum_{i=2}^{n-1}\sum_{k=1}^{m} r_i y_{ik} \Big/ \sum_{i=2}^{n-1} r_i \tag{20}$$

$\tau_{\min}$ and $\tau_{\max}$ are the lower and upper trail limits, respectively. The upper and lower trail limits are imposed to avoid stagnation. They are chosen as follows (Stützle & Hoos, 2000):

$$\tau_{\max} = \frac{F(s_{\text{gb}})}{(1-\rho)} \tag{21}$$

$$\tau_{\min} = \left(1 - \sqrt[n]{P_{\text{best}}}\right) \Big/ \left((\text{avg} - 1)\sqrt[n]{P_{\text{best}}}\right)\tau_{\max} \tag{22}$$

where avg is equal to $n/2$. P_{best} is the probability of constructing the best solution found when all the pheromone values have converged to either $\tau_{\min}$ or $\tau_{\max}$ (Levine & Ducatelle, 2004).

As suggested by Stützle and Hoos (2000), the pheromone values are initialized to the upper trail limit. This gives the ants a higher exploration ability in the early cycles. Another way of enhancing exploration is to reini-

Table 9
Results for data set 4

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p4.2.a	206	206	16.2	206	206	12.5	206	206	11.6	206	206	12.5
p4.2.b	341	338.7	20.3	341	338	16.5	341	338.7	15.8	341	340.2	16.7
p4.2.c	452	447.9	21.9	452	448.8	18.3	452	449.4	17.6	452	448	18.3
p4.2.d	531	527.5	23.5	531	528.7	19.8	530	528.4	19.1	531	528.2	19.9
p4.2.e	618	596.9	25.5	600	595.6	21.6	600	597	21	613	599.5	22.1
p4.2.f	687	672.6	27.5	672	667.6	23.8	672	663.8	23.3	672	664.9	24
p4.2.g	757	736.8	29.7	756	743.2	26	756	746.1	25.6	756	749.4	25.9
p4.2.h	827	818.2	31.6	819	812.8	27.4	819	812.4	26.8	820	815	27.3
p4.2.i	918	894.1	31.9	900	888.5	29.2	918	873.8	29.1	918	895.3	28.5
p4.2.j	965	953.2	34.6	962	949.6	30.4	962	945.2	30	962	960.2	30.7
p4.2.k	1022	1001.1	36.1	1016	1001.3	32.1	1016	1004.2	31.8	1016	1001.8	32.9
p4.2.l	1071	1063.5	37.5	1070	1062.2	33.7	1071	1058.9	33.6	1069	1060.8	34.4
p4.2.m	1130	1110.6	39.9	1115	1106.9	36.3	1119	1108.3	35.3	1113	1094.2	34.8
p4.2.n	1168	1146.9	41.3	1149	1133.6	36.1	1158	1148	37.9	1169	1146.6	38.3
p4.2.o	1215	1175.8	40.4	1209	1188	40.2	1198	1184.3	38.9	1210	1184.1	39.5
p4.2.p	1242	1215	43	1229	1211.7	39.9	1233	1206.9	38.3	1239	1206.3	39.2
p4.2.q	1263	1234.3	43.6	1253	1232.6	39.4	1252	1225.7	39.6	1260	1227.2	39.5
p4.2.r	1288	1263.4	45	1278	1257.5	42	1278	1261.6	41.4	1279	1264	42.3
p4.2.s	1304	1288.4	46	1304	1288.4	43.6	1303	1284.9	41.8	1304	1294.5	44.7
p4.2.t	1306	1304.4	47.1	1306	1305.1	42.7	1306	1303	42.9	1306	1306	44.2
p4.3.b	38	38	17.4	38	38	11.7	38	38	9.9	38	38	11.8
p4.3.c	193	193	22	193	193	16.3	193	193	14.8	193	193	16.4
p4.3.d	335	333	24.9	333	332.5	19.4	333	333	18.1	335	332	19.5
p4.3.e	468	463.2	28.3	468	465.6	22.3	468	466.4	21.1	468	465.6	22.6
p4.3.f	579	569.2	30.5	579	575.6	24.7	579	573.8	23.3	579	569	24.9
p4.3.g	653	651.6	31.6	652	652	26.1	653	647.2	25.2	652	649.4	26.3
p4.3.h	720	712.6	34.4	713	709.8	28.8	713	709.4	27.6	713	710.4	28.7
p4.3.i	796	779.2	36	793	778	30.2	793	781.9	29.7	786	775.6	30.4
p4.3.j	861	839.4	38.3	857	845.6	32	855	841.4	30.7	858	850.5	32.2
p4.3.k	918	895.7	38.5	913	900.7	33.6	910	899	32.6	910	896.6	33.4
p4.3.l	979	954.2	39.3	958	952.4	34.4	976	961.1	33.2	966	953.4	34
p4.3.m	1053	1023.1	41	1039	1019.8	35.8	1028	1003.4	35.1	1046	1028.8	36.7
p4.3.n	1121	1100.3	43.2	1109	1093.9	37.7	1112	1099.7	36.7	1103	1094.7	37.5
p4.3.o	1170	1158.1	44	1163	1154.2	37.6	1167	1155.6	37.2	1165	1157.6	37.9
p4.3.p	1221	1201.7	45.5	1202	1189.4	38.8	1207	1200.8	38.8	1207	1202.2	39.8
p4.3.q	1252	1227.4	46.8	1239	1232.8	41.3	1239	1221.8	39.4	1238	1231	41.6
p4.3.r	1267	1255.7	47.1	1263	1260.4	42.4	1263	1260.4	41.6	1263	1260.2	42.9
p4.3.s	1293	1283.7	48.3	1291	1284.9	43.3	1289	1282	42.4	1291	1286.2	43.5
p4.3.t	1305	1302.3	48.8	1304	1302.8	44.3	1303	1293.6	42.4	1304	1301.8	43.8
p4.4.d	38	38	22.4	38	38	14.9	38	38	12.2	38	38	15
p4.4.e	183	183	27.5	183	183	19.9	183	183	17.6	183	183	20.1
p4.4.f	324	324	30.2	324	323.5	22.9	324	322.2	20.7	324	323.5	23.1
p4.4.g	461	460.1	33.5	461	459.8	26.1	461	458.3	24	460	460	26.2
p4.4.h	571	552	35.5	556	556	28.1	556	555.2	26.9	556	554.2	28.6
p4.4.i	657	641.6	37.5	653	642.6	30.3	652	643.6	28.4	653	649.1	30.8
p4.4.j	732	726.7	39.5	731	721.2	32.3	711	707.3	30	731	726.8	32.8
p4.4.k	821	814.2	40.9	820	815.3	34.1	818	813	32.2	818	814	33.7
p4.4.l	880	868.4	42.8	877	871.5	35.6	875	870.2	33.8	875	870.3	35.7
p4.4.m	918	904.7	43.4	911	909.1	36.5	906	903.1	34.9	911	906.9	36.9
p4.4.n	961	946.3	44.4	956	948.9	37.4	956	948	36.3	956	952.3	37.7
p4.4.o	1036	1001.1	45.9	1030	1012.3	39.2	1021	1002.7	37.7	1029	1015.5	39.6
p4.4.p	1111	1074	47	1108	1073.5	41	1088	1064.4	39.7	1110	1099.4	42.2
p4.4.q	1145	1106.2	47.5	1150	1117.2	41.1	1137	1107.7	39.8	1148	1122.5	41.5
p4.4.r	1200	1168.7	49.2	1195	1153	41.7	1195	1163.2	40.9	1194	1161.2	42.9
p4.4.s	1249	1233.9	50.2	1256	1229.2	42.8	1249	1213.7	41.9	1252	1238.1	43.8
p4.4.t	1281	1268.4	51.1	1281	1276.2	44.2	1283	1273.5	43.2	1281	1268.6	44.3

Table 10
Results for data set 5

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p5.2.b	20	20	7	20	20	5.3	20	20	4.7	20	20	5.3
p5.2.c	50	50	7.9	50	50	6.2	50	50	5.7	50	50	6.2
p5.2.d	80	80	9	80	80	7.3	80	80	6.8	80	80	7.4
p5.2.e	180	180	9.6	180	180	7.9	180	180	7.4	180	180	8
p5.2.f	240	240	10.1	240	240	8.4	240	240	7.9	240	240	8.5
p5.2.g	320	320	10.8	320	320	9.2	320	320	8.7	320	320	9.2
p5.2.h	410	404.5	11.6	410	402.5	10.1	410	403	9.6	410	403.5	10.1
p5.2.i	480	480	12.3	480	480	10.7	480	480	10.3	480	480	10.8
p5.2.j	580	580	12.9	580	580	11.3	580	580	10.9	580	580	11.4
p5.2.k	670	670	13.7	670	669.5	12.2	670	669.5	11.8	670	670	12.2
p5.2.l	800	778	14.5	800	773	12.8	800	773	12.4	800	774	13
p5.2.m	860	859.5	15.1	860	859.5	13.5	860	859	13.1	860	860	13.6
p5.2.n	925	921	15.7	920	919	14.1	920	920	13.7	925	920.5	14.2
p5.2.o	1020	1011	16.3	1020	1012	14.7	1010	1010	14.3	1010	1010	14.8
p5.2.p	1150	1143.5	16.9	1150	1150	15.4	1150	1150	15.1	1150	1150	15.5
p5.2.q	1195	1194	17.7	1195	1192.5	16.1	1195	1193	15.8	1195	1195	16.3
p5.2.r	1260	1258.5	18.3	1260	1257.5	16.9	1260	1259	16.5	1260	1256.5	16.9
p5.2.s	1340	1324	19.1	1330	1325	17.5	1330	1323.5	17.3	1330	1324	17.7
p5.2.t	1400	1382	19.7	1400	1377	18.2	1400	1379.5	18	1400	1382	18.6
p5.2.u	1460	1452.5	20.5	1460	1447	19.1	1460	1457.5	18.7	1460	1448	19.1
p5.2.v	1505	1491.5	21.1	1495	1487	19.5	1500	1496.5	19.4	1495	1486.5	19.7
p5.2.w	1560	1537.5	21.7	1555	1541.5	20.4	1555	1549.5	20.2	1555	1536	20.1
p5.2.x	1610	1595.5	22.3	1610	1586.5	20.8	1610	1607	20.9	1610	1593.5	21
p5.2.y	1645	1631.5	22.6	1645	1633.5	21.4	1645	1631.5	21.3	1645	1632	21.4
p5.2.z	1680	1672.5	23	1680	1680	21.8	1680	1673	21.6	1680	1677	21.8
p5.3.b	15	15	9.7	15	15	7.1	15	15	6.1	15	15	7.2
p5.3.c	20	20	9.6	20	20	7.4	20	20	6.2	20	20	7.4
p5.3.d	60	60	10.9	60	60	8.3	60	60	7.3	60	60	8.3
p5.3.e	95	95	11.7	95	95	9.2	95	95	8.3	95	95	9.3
p5.3.f	110	110	12.5	110	110	10	110	110	9	110	110	10
p5.3.g	185	185	12.8	185	185	10.3	185	185	9.3	185	185	10.3
p5.3.h	260	260	13.9	260	260	11.4	260	260	10.5	260	260	11.5
p5.3.i	335	335	14.3	335	335	11.8	335	335	10.9	335	335	11.9
p5.3.j	470	470	15.1	470	470	12.6	470	470	11.8	470	470	12.7
p5.3.k	495	495	15.6	495	495	13.2	495	495	12.4	495	495	13.3
p5.3.l	595	590	16.3	595	586	13.9	595	584	13.1	595	584	14
p5.3.m	650	649.5	17	650	649.5	14.6	650	649	13.8	650	649	14.7
p5.3.n	755	755	17.6	755	755	15.3	755	755	14.6	755	755	15.4
p5.3.o	870	865	18.3	870	864.5	15.9	870	867.5	15.2	870	864	16
p5.3.p	990	990	18.8	990	990	16.5	990	989	15.8	990	990	16.6
p5.3.q	1070	1061.5	19.5	1065	1056.5	17.2	1065	1057.5	16.5	1065	1056	17.2
p5.3.r	1125	1114.5	20	1120	1113	17.7	1125	1114.5	17.1	1125	1114.5	17.8
p5.3.s	1190	1187	20.7	1190	1180.5	18.4	1190	1178.5	17.8	1185	1179	18.5
p5.3.t	1260	1251	21.2	1250	1246.5	18.9	1255	1246.5	18.3	1260	1250.5	19
p5.3.u	1345	1336	21.6	1330	1319	19.8	1335	1320	18.9	1335	1326	19.6
p5.3.v	1425	1402	22.1	1425	1412.5	20.2	1425	1414.5	19.8	1420	1398.5	20.1
p5.3.w	1485	1458	22.7	1465	1455	20.5	1465	1452	20.1	1465	1452.5	20.6
p5.3.x	1540	1513.5	23.1	1535	1523.5	21.3	1540	1518	20.7	1540	1522	21.2
p5.3.y	1590	1555	23.5	1590	1552.5	21.4	1590	1547.5	20.8	1590	1552.5	21.4
p5.3.z	1635	1610	23.8	1635	1616.5	21.7	1635	1623	21.4	1635	1615.5	21.8
p5.4.c	20	20	12.6	20	20	9.2	20	20	7.7	20	20	9.3
p5.4.d	20	20	12.2	20	20	9.3	20	20	7.7	20	20	9.4
p5.4.e	20	20	12.4	20	20	9.4	20	20	7.7	20	20	9.4
p5.4.f	80	80	14.4	80	80	11.1	80	80	9.6	80	80	11.1
p5.4.g	140	140	15.3	140	140	11.9	140	140	10.5	140	140	12
p5.4.h	140	140	15.7	140	140	12.5	140	140	11.1	140	140	12.5

Table 10 (*continued*)

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p5.4.i	240	240	16.3	240	240	13	240	240	11.7	240	240	13.1
p5.4.j	340	340	17.1	340	340	13.8	340	340	12.6	340	340	13.9
p5.4.k	340	340	17.8	340	340	14.5	340	340	13.3	340	340	14.6
p5.4.l	430	429.5	18	430	428	14.9	430	428	13.7	430	428	15
p5.4.m	555	554	18.9	555	552	15.7	555	551.5	14.5	555	553	15.8
p5.4.n	620	620	19.2	620	620	16.1	620	620	15	620	620	16.2
p5.4.o	690	690	19.9	690	689.5	16.9	690	690	15.8	690	689.5	16.9
p5.4.p	765	758	20.4	760	755	17.4	760	752	16.3	760	753	17.4
p5.4.q	860	851	21.2	860	837.5	18	860	847	17.2	860	839.5	18.2
p5.4.r	960	960	21.7	960	960	18.8	960	958	17.8	960	954	18.8
p5.4.s	1030	1020	22.3	1030	1017	19.2	1030	1019.5	18.3	1030	1011.5	19.3
p5.4.t	1160	1152	22.7	1160	1134.5	19.7	1160	1139.5	18.8	1160	1131	19.8
p5.4.u	1300	1300	23	1300	1274.5	20.2	1300	1260	19.3	1300	1282.5	20.3
p5.4.v	1320	1320	23.5	1320	1292.5	20.7	1320	1297	19.9	1320	1300.5	20.8
p5.4.w	1390	1373.5	24	1380	1374	21.1	1390	1374.5	20.3	1380	1374.5	21.2
p5.4.x	1450	1443	24.4	1450	1440.5	21.5	1450	1439	20.8	1450	1441	21.6
p5.4.y	1520	1513	24.8	1510	1483	21.9	1510	1492	21.4	1500	1485	22.1
p5.4.z	1620	1585.5	25.2	1620	1567	22.5	1575	1549	21.7	1580	1553.5	22.5

tialize the pheromone trails to the upper trail limit. In our algorithm, we reinitialize the pheromone trails once no better solution can be found for N_{ni} cycles.

3.4. Local search

It is known that the coupling of ACO and local search is effective to improve the performance of the ACO (e.g., Levine & Ducatelle, 2004; Solnon, 2002; Solnon & Fenet, 2006). In fact, ACO performs a rather coarse-grained search, and the solutions constructed can then be locally optimized by an adequate local search procedure (Dorigo & Stützle, 2002).

The local search procedure we used is based on (Chao et al., 1996a). The main procedure (procedure **LocalSearch** in Fig. 1) is as follows: each path is shortened by using a 2-opt procedure and then inserted as many feasible points as possible. This local search procedure is iterated until no improvement can be obtained.

4. Experimental results

We now experimentally study the performance of ACO–TOP. The algorithm was coded in C++ and tested on a PC with 3.0 GHZ Intel CPU. The computational experiments have been made on a set of 387 benchmark instances taken from (Chao et al., 1996a). These instances are included in seven sets. The numbers of vertices are 32, 21, 33, 100, 66, 64 and 102, respectively. The coordinate and reward of each vertex is identical in all instances of the same set. In each set, there are three groups which have different numbers of vehicles. An instance in each group is characterized by a different value of $T_{\max}$.

4.1. Parameter settings

Before studying our algorithm, we have to identify a good parameter setting. We have studied the influence of these parameters on the basis of experimental results. For each instance, 10 independent tests were carried out. In all experiments, we used a mixed strategy to schedule s_{gb} and s_{ib} for pheromone updating: every 5 cycles, s_{gb} is used for updating, while s_{ib} is used in other cycles. Since this paper aims to propose a fast and effective algorithm, the maximal number of cycles N_C was set to 2000. The number of ants n_a was set to 20. $N_{ni} = 250$. As chosen in many other applications (Stützle & Hoos, 2000), $\alpha = 1$, $\rho = 0.98$, $P_{\text{best}} = 0.05$. In the preliminary experiments, we observed that β and γ are crucial to the performance of ACO–TOP.

Table 11
Results for data set 6

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p6.2.d	192	189	8.8	192	186.6	7.2	192	188.4	6.8	192	188.4	7.3
p6.2.e	360	359.4	10.3	360	358.8	8.8	360	357	8.4	360	360	8.9
p6.2.f	588	587.4	11.9	588	585.6	10.5	588	585	9.9	588	586.8	10.5
p6.2.g	660	660	12.9	660	660	11.3	660	660	11	660	660	11.4
p6.2.h	780	780	14	780	780	12.5	780	780	12.1	780	780	12.6
p6.2.i	888	888	15	888	888	13.5	888	888	13.1	888	888	13.6
p6.2.j	948	947.4	15.9	948	948	14.5	948	948	14.1	948	948	14.5
p6.2.k	1032	1032	16.9	1032	1032	15.4	1032	1032	15.2	1032	1032	15.5
p6.2.l	1116	1111.2	17.9	1110	1106.4	16.5	1116	1111.2	16.5	1116	1110.6	16.6
p6.2.m	1188	1184.4	19.1	1188	1175.4	17.6	1188	1182.6	17.5	1188	1183.8	17.8
p6.2.n	1260	1230.6	19.6	1260	1234.8	18.4	1254	1230.6	18.1	1260	1235.4	18.3
p6.3.g	282	278.4	12.6	282	277.2	10.2	282	276.6	9.4	282	277.8	10.2
p6.3.h	444	427.8	13.5	444	427.8	11.2	438	428.4	10.7	438	430.2	11.4
p6.3.i	642	640.8	15.3	642	640.8	13.1	642	639.6	12.3	642	638.4	13.2
p6.3.j	828	825.6	16.4	828	825	14.2	828	825	13.5	828	825.6	14.2
p6.3.k	894	888.6	17.5	888	888	15.2	888	888	14.5	894	888.6	15.3
p6.3.l	1002	996	18.5	1002	996	16.4	1002	993	15.7	1002	996	16.4
p6.3.m	1080	1071.6	19.4	1074	1069.8	17.1	1080	1071.6	16.8	1080	1074	17.4
p6.3.n	1170	1159.2	20.3	1164	1160.4	18.2	1164	1155	17.7	1164	1159.8	18.3
p6.4.j	366	363	16.1	366	362.4	12.9	366	361.8	11.7	366	363	13.1
p6.4.k	528	525	16.9	528	522	14	528	517.8	12.8	528	520.2	14
p6.4.l	696	671.4	18.1	696	675.6	15.2	696	679.2	14.3	696	674.4	15.3
p6.4.m	912	885.6	19.5	912	873.6	16.6	912	876.6	15.8	912	886.2	16.7
p6.4.n	1068	1061.4	20.3	1068	1062	17.4	1068	1062	16.6	1068	1066.2	17.5

The values tested for β and γ were $\{0, 0.25, 0.5, 1, 2, 4, 8\}$. Experimental results show that 0.5 is a good choice for these two parameters.

4.2. The comparative study on four construction methods

In Table 1, the experimental results on the 21 groups are shown. The best rewards, the average rewards and the computational times (average over all instances in each group) are given for each method. It can be noticed that the sequential method finds the largest rewards and the simultaneous method is better than the deterministic-concurrent and random-concurrent methods. In terms of the computational time, the random-concurrent method performs faster than the other three. However, the sequential method can solve each instance within 51.1 s (on average). Therefore, the sequential method is an excellent compromise between solution quality and computational time. In the following, we only discuss the sequential method.

4.3. Experimental comparison with several existing algorithms

Our algorithm has been compared with several existing algorithms on the same instances:

CGW: the five-step heuristic proposed by Chao et al. (1996a);
TMH: the tabu search heuristic proposed by Tang and Miller-Hooks (2005);
GTP: the tabu search with the penalty strategy proposed by Archetti et al. (2007);
GTF: the tabu search with the feasible strategy proposed by Archetti et al. (2007);
FVF: the fast variable neighborhood search proposed by Archetti et al. (2007);
SVF: the slow variable neighborhood search proposed by Archetti et al. (2007).[1] It differs from FVF on the parameter setting.

[1] The results of GTP, GTF, FVF and SVF are available at www-c.eco.unibs.it/'archetti/TOP.zip.

Table 12
Results for data set 7

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p7.2.a	30	30	12.6	30	30	8.5	30	30	7.6	30	30	8.6
p7.2.b	64	64	13.6	64	64	9.5	64	64	8.6	64	64	9.5
p7.2.c	101	101	14.9	101	101	10.9	101	101	10	101	101	10.9
p7.2.d	190	190	16.8	190	190	12.8	190	190	12	190	190	12.9
p7.2.e	290	290	18.7	290	290	14.7	290	290	13.9	290	290	14.8
p7.2.f	387	386.7	20.5	387	386.7	16.5	387	387	15.7	387	386.4	16.6
p7.2.g	459	459	21.8	459	459	17.8	459	459	17	459	459	17.8
p7.2.h	521	521	23.1	521	520.6	19	521	521	18.6	521	521	19.3
p7.2.i	580	578.6	24.5	579	578.3	20.5	579	578.3	19.7	579	578.3	20.6
p7.2.j	646	644	26.1	646	644.6	22.3	646	645.7	21.4	646	644.3	22.1
p7.2.k	705	701.2	26.9	704	701.8	22.9	704	702.8	22.2	704	702.8	23
p7.2.l	767	765.4	27.8	767	765.5	23.9	767	766.5	23.1	767	764.3	24.1
p7.2.m	827	827	29.5	827	824.5	25.9	827	825.8	25.3	827	826.4	25.7
p7.2.n	888	878	30.6	878	878	27.2	878	878	26.6	878	877.4	27.1
p7.2.o	945	940.1	32.7	945	935.8	28.5	940	933.2	28.4	941	935.1	29.1
p7.2.p	1002	991.3	33.9	991	983.5	30.4	993	985.4	29.6	993	986.6	29.5
p7.2.q	1043	1040	35.7	1042	1038.6	31.6	1043	1036.5	31.1	1043	1033.4	30.9
p7.2.r	1094	1078.9	36.5	1093	1082.9	32.9	1088	1075.6	31.1	1094	1084.4	32.7
p7.2.s	1136	1115.2	36.9	1136	1118.3	33.7	1134	1119.2	32.9	1131	1115.7	32.8
p7.2.t	1179	1146.6	36.4	1179	1161.9	34.9	1179	1153.1	34.2	1179	1157.1	34.4
p7.3.b	46	46	18.5	46	46	12.2	46	46	10.6	46	46	12.3
p7.3.c	79	79	19.3	79	79	13.2	79	79	11.5	79	79	13.2
p7.3.d	117	117	20.8	117	117	14.5	117	117	13	117	117	14.6
p7.3.e	175	175	22.4	175	175	16.1	175	175	14.7	175	175	16.2
p7.3.f	247	247	24.1	247	247	17.9	247	247	16.5	247	247	18
p7.3.g	344	344	25.6	344	344	19.5	344	344	18.1	344	344	19.6
p7.3.h	425	424.3	27.7	425	423.9	21.6	425	423.1	20.3	425	424.5	21.7
p7.3.i	487	485.3	29.3	487	485.1	23.5	486	485.6	22.2	487	485	23.4
p7.3.j	564	563.2	30.7	564	562.8	24.8	564	563.4	24	564	563.3	25
p7.3.k	633	629.5	31.5	632	627.1	25.6	633	629.4	24.8	633	629.6	26
p7.3.l	684	680.7	33.1	683	680.5	27.4	684	679	26.1	684	681.2	27.4
p7.3.m	762	759.1	33.8	762	756.3	28.1	762	754.2	27.2	762	755.5	28.3
p7.3.n	820	813.9	35.1	819	811	29.2	819	811.2	28.2	820	813	29.6
p7.3.o	874	874	35.7	874	873.7	30.1	874	873	29.1	874	873.7	30.1
p7.3.p	929	925.6	36.8	925	922.3	31.1	926	924.1	30.2	925	923.6	31.3
p7.3.q	987	984.5	38	987	983.1	32.5	987	982.5	31.5	987	981	32.6
p7.3.r	1026	1018.4	39.6	1024	1017	34.2	1021	1015	33.2	1022	1016.4	34.3
p7.3.s	1081	1070.3	41.3	1081	1062.2	35.1	1081	1062.6	34.3	1077	1061.5	35.4
p7.3.t	1118	1107.2	42.3	1117	1101	35.5	1103	1086.5	34.1	1117	1108	36.7
p7.4.b	30	30	23	30	30	14.8	30	30	12.3	30	30	14.9
p7.4.c	46	46	23.6	46	46	15.4	46	46	12.9	46	46	15.5
p7.4.d	79	79	24.5	79	79	16.6	79	79	14.1	79	79	16.6
p7.4.e	123	123	26.1	123	123	18	123	123	15.7	123	123	18.1
p7.4.f	164	164	27.9	164	164	19.6	164	164	17.3	164	164	19.7
p7.4.g	217	217	29.1	217	217	20.9	217	217	18.8	217	217	21.1
p7.4.h	285	285	30.7	285	285	22.7	285	285	20.6	285	285	22.7
p7.4.i	366	366	32.1	366	366	24.2	366	366	22.1	366	366	24.3
p7.4.j	462	462	33.9	462	461.7	26.3	462	461.1	24.2	462	462	26.3
p7.4.k	520	518	35.3	520	517.2	27.8	520	461.1	25.8	520	462	28
p7.4.l	590	581.7	37	590	580.5	29.2	590	517.8	27.5	590	517.9	29.5
p7.4.m	646	643.9	37.9	644	642.9	30.2	646	583.6	28.6	646	584.8	30.4
p7.4.n	730	725.6	39.4	725	724.4	31.6	725	643.4	29.9	726	642.2	31.7
p7.4.o	781	777.5	40.2	778	775.2	32.8	781	724.4	31.2	778	724.5	33.1
p7.4.p	846	839.4	41.3	846	838.7	34.8	838	776.2	32.7	842	776.5	34.4
p7.4.q	909	905.1	42.1	909	905.6	34.8	909	832.9	33.5	909	835.5	35
p7.4.r	970	969.2	42.5	970	968.8	36	970	904.1	34.3	970	904.2	35.5

(*continued on next page*)

Table 12 (*continued*)

Problem name	Sequential			Deterministic-concurrent			Random-concurrent			Simultaneous		
	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time	Best	Aver	Time
p7.4.s	1022	1017.7	44	1019	1014.8	36.6	1021	968.4	35.2	1019	966.5	36.8
p7.4.t	1077	1072.8	44.7	1072	1070.5	37.6	1077	1014.5	36.8	1077	1013.4	38.1

Since only the best rewards of these algorithms are available, we report the best rewards of all algorithms.[2] When comparing these algorithms, one problem is computational precision of the final path. It is clear that the final path of CGW is rounded to one decimal place (Chao et al., 1996a). However, the computational precision of other algorithm is implicit. In fact, we found that 19 instances have different best rewards when different kinds of precision are considered. In order to compare, we report the results of ACO–TOP without rounding here. In Appendix A, we give new best rewards obtained when the final path is rounded to one decimal place.

For each algorithm, the (average) best reward of each group is reported in Table 2. It can be seen that ACO–TOP and SVF obtain the most promising rewards. Each of them performs better than other algorithms in 15 groups. Moreover, ACO–TOP finds the best-so-far rewards for 347 problems and new best rewards for 12 problems. The new best rewards are shown in Table 3. The column "problem name" indicates the name of each problem. Each problem is denoted by the notation p$x{\cdot}y{\cdot}z$, where x represents the set to which the problem belongs, y indicates the number of vehicles and z is a label that differentiates between problems from the same set and with the same number of vehicles. According to these tables, ACO–TOP has a promising ability to find good solutions.

Table 4 shows the average computational time required by each algorithm. The CGW algorithm was run on a SUN 4/730 Workstation 25 MHz, TMH was run on a DEC Alpha XP1000 computer 667 MHz, while the other four algorithms were run on a personal computer with 2.8 GHZ CPU. Since our algorithm requires less than one minute on each instance, ACO–TOP can find good solutions in a reasonable amount of time.

5. Conclusions

In this paper, an ACO approach, called ACO–TOP, is proposed for the team orienteering problem. We proposed the sequential, deterministic-concurrent and random-concurrent and simultaneous methods to construct solutions. Moreover, we tested them on classical benchmark problems. The experimental results show that the sequential method can obtain the best solution quality within less than one minute on each instance. Finally, ACO–TOP was compared to several promising algorithms. The experimental results on the benchmark problems show that ACO–TOP can compete efficiently and effectively with these algorithms.

Acknowledgements

The authors thank the anonymous reviewers for their helpful comments and suggestions. This work is supported by the National Natural Science Foundation of China, No. 60475023 and the Ph. D. Programs Foundation of Ministry of Education of China, No. 20050698032.

Appendix A

Table 5 shows the new best rewards obtained when the final length is rounded to one decimal place.

The detailed results for set 1–set 7 are given in Tables 6–12.

[2] A detailed table of results for all the test instances, together with the instances themselves, can be obtained by email or see Appendix A.

References

Archetti, C., Hertz, A., & Speranza, M. G. (2007). Metaheuristics for the team orienteering problem. *Journal of Heuristics, 13*, 49–76.

Ballou, R., & Chowdhury, M. (1980). MSVS: An extended computer model for transport mode selection. *The Logistics and Transportation Review, 16*, 325–338.

Diaby, M., & Ramesh, R. (1995). The distribution problem with carrier service: A dual based penalty approach. *ORSA Journal on Computing, 7*, 24–35.

Blum, C., & Dorigo, M. (2004). The hyper-cube framework for ant colony optimization. *IEEE Transactions on System Man, and Cybernetics – Part B, 34*, 1161–1172.

Boussier, S., Feillet, D., Gendreau, M. (2006). An exact algorithm for team orienteering problems. 4OR.

Butt, S. E., & Cavalier, T. M. (1994). A Heuristic for the multiple path maximum collection problem. *Computers and Operations Research, 21*, 101–111.

Butt, S., & Ryan, D. (1999). An optimal solution procedure for the multiple path maximum collection problem using column generation. *Computer and Operations Research, 26*, 427–441.

Chao, I-M., Golden, B., & Wasil, E. A. (1996a). The team orienteering problem. *European Journal of Operational Research, 88*, 464–474.

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

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on System Man, and Cybernetics–Part B, 26*, 29–41.

Dorigo, M., Di Caro, G., & Gambardella, L. M. (1999). Ant algorithms for distributed discrete optimization. *Artificial Life, 5*, 137–172.

Dorigo, M., & Stützle, T. (2002). The ant colony optimization metaheuristic: Algorithms, applications, and advances. In F. Glover & G. Kochenberger (Eds.), *Handbook of metaheuristics* (pp. 251–285). Norwell, MA, USA: Kluwer Academic Publishers..

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization: MA*. Cambridge: MIT Press.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science, 344*, 243–278.

Golden, B., Levy, L., & Vohra, R. (1987). The orienteering problem. *Naval Research Logistics, 34*, 307–318.

Hall, R., & Racer, M. (1995). Transportation with common carrier and private fleets: System assignment and shipment frequency optimization. *IIE Transactions, 27*, 217–225.

Levine, J., & Ducatelle, F. (2004). Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society, 55*, 705–716.

Solnon, C. (2002). Ants can solve constraint satisfaction problems. *IEEE Transaction on Evolutionary Computation, 6*, 347–357.

Solnon, C., & Fenet, S. (2006). A study of ACO capabilities for solving the maximum clique problem. *Journal of Heuristics, 12*, 155–180.

Stützle, T., & Hoos, H. H. (2000). Max–min ant system. *Future Generation Computer Systems, 16*, 889–914.

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

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *Journal of the Operational Research Society, 35*, 797–809.