Soft Comput (2012) 16:597–614
DOI 10.1007/s00500-011-0759-3

ORIGINAL PAPER

Multi-objective ant colony optimization based on decomposition for bi-objective traveling salesman problems

Jixang Cheng · Gexiang Zhang · Zhidan Li · Yuquan Li

Published online: 16 September 2011

Abstract This paper proposes a framework named multi-objective ant colony optimization based on decomposition (MoACO/D) to solve bi-objective traveling salesman problems (bTSPs). In the framework, a bTSP is first decomposed into a number of scalar optimization subproblems using Tchebycheff approach. To suit for decomposition, an ant colony is divided into many subcolonies in an overlapped manner, each of which is for one subproblem. Then each subcolony independently optimizes its corresponding subproblem using single-objective ant colony optimization algorithm and all subcolonies simultaneously work. During the iteration, each subproblem maintains an aggregated pheromone trail and an aggregated heuristic matrix. Each subcolony uses the information to solve its corresponding subproblem. After an iteration, a pheromone trail share procedure is evoked to realize the information share of those subproblems solved by common ants. Three MoACO algorithms designed by, respectively, combining MoACO/D with AS, MMAS and ACS are presented. Extensive experiments conducted on ten bTSPs with various complexities manifest that MoACO/D is both efficient and effective for solving bTSPs and the ACS version of MoACO/D outperforms three well-known MoACO algorithms on large bTSPs according to several performance measures and median attainment surfaces.

Keywords Decomposition · Multi-objective · Ant colony optimization · Bi-objective traveling salesman problems

1 Introduction

In real-world applications, there are lots of multi-objective combinatorial optimization problems, where several conflicting objectives have to be optimized simultaneously and the goal is to find a set of solutions with good trade-offs among different objectives, usually called Pareto solutions (Miettinen 1999). Inspired by the foraging behavior of real ants, Dorigo proposed a novel metaheuristic algorithm, ant colony optimization (ACO), to solve classical traveling salesman problems (TSPs) (Dorigo et al. 1996; Dorigo and Stützle 2004). Subsequently various versions of ACO were designed, such as ant colony system (ACS) (Dorigo and Gambardella 1997), Max–Min ant system (MMAS) (Stützle and Hoos 2000) and rank-based ant system (ASrank) (Bullnheimer et al. 1999). As an excellent technique for solving single-objective combinatorial optimization problems, ACO is also suitable for multi-objective combinatorial optimization problems, as it is a colony-based optimization approach which can obtain a certain number of Pareto solutions in a single run (García-Martínez et al. 2007; Angus and Woodward 2009).

In the past two decades, various versions of multi-objective ACO (MoACO) were presented (Mariano and Morales 1999; Iredi et al. 2001; T'kindt et al. 2002; Barán and Schaerer 2003; Doerner et al. 2006; López-Ibáñez and Stützle 2010a; Cardoso et al. 2011). In García-Martínez et al. (2007), a taxonomy of MoACO was discussed in terms of the number of pheromone trails and heuristic matrices. Also a comparative analysis was made using

J. Cheng · G. Zhang (✉) · Y. Li
School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, China
e-mail: zhgxdylan@126.com

Z. Li
School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China

experiments carried out on bi-objective traveling salesman problems (bTSPs) and some guidelines on how to devise MoACO algorithm were given. Angus and Woodward (2009) summarized existing MoACO algorithms and provided a classification method, which is an extension of the ACO framework in Gambardella et al. (1999). In what follows, we group various MoACO algorithms into two main classes. In the first class, several objectives were dealt with in a lexicographic ordering manner, such as in MOAQ (Mariano and Morales 1999) and SACO (T'kindt et al. 2002). This class requires the prior knowledge of problems, which could hardly be obtained in real-world applications. The second class of MoACO simultaneously handles several objectives by the use of multiple pheromone trails (Iredi et al. 2001), multiple heuristic matrices (Barán and Schaerer 2003) or a problem-specific pheromone update scheme (Doerner et al. 2006). When designing multi-objective algorithms, two dilemmas are encountered, i.e., (1) a good balance among conflicting objectives is difficult to maintain in the process of iteration, which results in an uneven Pareto front; and (2) it is difficult to approach the Pareto optimal front. These predicaments can be narrowed to the problem of how to appropriately configure the algorithm. Recently, López-Ibáñez and Stützle (2010a) proposed a software framework, which allowed to instantiate the most prominent MoACO algorithms, and the authors also applied automatic algorithm configuration techniques to MoACO algorithm.

It is well known that a Pareto solution to a multi-objective optimization problem (MOP) could be an optimal solution of a scalar optimization problem whose objective is an aggregation of all objectives in the MOP (Miettinen 1999). To obtain a set of Pareto solutions, a MOP can be decomposed into a number of scalar optimization subproblems and thus single-objective problem solvers can be used without many modifications. In Zhang and Li (2007) and Peng et al. (2009), the idea was introduced into a multi-objective genetic algorithm and better results than NSGA-II (Deb et al. 2002) and MOGLS (Ishibuchi and Murata 1998; Jaszkiewicz 2002) were obtained by testing multi-objective 0–1 knapsack problems, multi-objective continuous optimization problems and/or bTSPs. To the best of our knowledge, no investigation focuses on using the decomposition idea to enhance the MoACO performance.

In this paper, a framework named multi-objective ant colony optimization based on decomposition (MoACO/D) is proposed to solve bTSPs. To use the decomposition idea to specify MoACO, a bTSP is first decomposed into a number of scalar optimization subproblems, each of which maintains an aggregated pheromone trail and an aggregated heuristic matrix. In order to adapt the decomposition, an ant colony is divided into subcolonies in an overlapped way, each of which is for one subproblem. During the iteration, each subcolony uses the aggregated pheromone trail and the aggregated heuristic matrix to solve its corresponding subproblem. After an iteration, a pheromone trail share procedure is evoked to realize the information share of those subproblems solved by common ants. It is worth noting that each subcolony independently optimizes its corresponding subproblem using single-objective ACO algorithm and all subcolonies simultaneously work in a single iteration. Based on the framework, three MoACO algorithms designed by, respectively, combining MoACO/D with AS, MMAS and ACS are presented. In the experiments, two important parameters in every MoACO/D variation are discussed in an empirical way. A large number of experiments are conducted on ten bTSPs with various complexities and experimental results show that MoACO/D is efficient and effective for solving bTSPs and the ACS version of MoACO/D obtains the best performance in eight out of ten bTSPs in terms of several measures and median attainment surfaces.

2 Problem statement

As an extension of a traditional TSP, a bTSP is to seek for paths simultaneously satisfying two conflicting objectives. As usual, a bTSP can be modeled as an undirectedly weighted graph $G(L, A)$, where L is a set of vertices and A is a set of undirected arcs linking each pair of vertices. Thus a solution π to a bTSP is a permutation of L vertices. A bTSP can be formulated as

$$\begin{cases} \min f_1(\pi) = \sum_{i=1}^{L-1} c^1_{\pi(i)\pi(i+1)} + c^1_{\pi(L)\pi(1)} \\ \min f_2(\pi) = \sum_{i=1}^{L-1} c^2_{\pi(i)\pi(i+1)} + c^2_{\pi(L)\pi(1)}, \end{cases} \tag{1}$$

where $f_1(\pi)$ and $f_2(\pi)$ are two objective functions; $c^m_{\pi\ (i)\pi\ (j)}$ is the cost value of the arc (i, j) connecting the vertices i and j in the mth objective, $m = 1, 2, i, j \in \{1, 2, \ldots, L\}$ and $i \neq j$. The cost could be related to distance, time, money or energy.

In the theory of computational complexity, a bTSP belongs to the class of NP-complete problem. Lots of engineering problems, such as multi-objective network structure design problems, multi-objective machine scheduling problems and multi-objective vehicle routing problems, can be formulated as bTSPs. Consequently, a bTSP is frequently employed as a benchmark problem to evaluate the performance of multi-objective optimization algorithms (García-Martínez et al. 2007; Peng et al. 2009; López-Ibáñez and Stützle 2010b; Jaszkiewicz and Zielniewicz 2009).

3 MoACO/D

This section starts from the decomposition approach and then turns to the detailed description of MoACO/D framework. Finally, the characteristics of MoACO/D and a contrast with existing approaches are presented. For the convenience of discussion, a bTSP is taken as an example in this section.

3.1 Decomposition

This section discusses the decomposition method, the Tchebycheff approach, which will be used in MoACO/D to convert a bTSP into a number of scalar optimization subproblems. The Tchebycheff approach can be described as follows (Miettinen 1999): Let $\{\boldsymbol{\lambda}^1, \boldsymbol{\lambda}^2, \ldots, \boldsymbol{\lambda}^N\}$ and $\boldsymbol{z}^* = (z_1^*, z_2^*)$ denote a set of uniform spread weighted vectors and a reference point, respectively, where $z_i^* = \min\{f_i(\pi)|\pi \in \Omega\}, i = 1, 2$; then a bTSP can be decomposed into N scalar optimization subproblems and MoACO/D minimizes all these N subproblems simultaneously in a single run. The objective function of the kth subproblem is depicted as

$$\min g_k(\pi|\boldsymbol{\lambda}^k, \boldsymbol{z}^*) = \max_{1 \le m \le 2} \{\lambda_m^k |f_m(\pi) - z_m^*|\}, \qquad (2)$$

where $\boldsymbol{\lambda}^k = (\lambda_1^k, \lambda_2^k)$ is the weight vector of the kth subproblem, $k = 1, 2, \ldots, N$; π is a feasible solution of a bTSP.

If π^* is a Pareto solution of a bTSP, there exists a weight vector $\boldsymbol{\lambda}$ such that π^* is the optimal solution of (2). On the other hand, given a weight vector $\boldsymbol{\lambda}$, the optimal solution of (2) is a Pareto solution of the bTSP. Thus with a set of evenly spread weight vectors and a good solver for (2), we can obtain Pareto solutions having good approximations and distributions.

3.2 MoACO/D

Given N evenly spread weight vectors, MoACO/D first decomposes a bTSP into N scalar optimization subproblems, and an ant colony with N ants is divided into N subcolonies according to the distance between vectors. Meanwhile, each subproblem maintains an aggregated pheromone trail and an aggregated heuristic matrix, which are initialized before the process of iteration and never aggregated again during the run. MoACO/D also uses an external unbounded archive to store non-dominated solutions. During the iteration, each subcolony independently and simultaneously optimizes its subproblem using corresponding pheromone trail and heuristic matrix. After an iteration, a pheromone trail share procedure is evoked to implement the information share of those subproblems solved by common ants, and solutions stored in the archive are updated. The pseudocode for MoACO/D is shown in Fig. 1, where each step is described as follows:

```
    Begin
      $t \leftarrow 1$
(i)  Initialization
        a) Decompose bTSP
        b) Divide ant colony
        c) Initialize pheromone trails and heuristic matrices
(ii) While (not termination condition) do
        For $k = 1, \cdots, N$
          a) Construct tours
          b) Update pheromone trails
          c) Update reference point
          d) Share pheromone trails
          e) Update archive
        End for
        $t \leftarrow t + 1$
     End while
    End Begin
```

Fig. 1 The pseudocode for MoACO/D

(1) This step consists of three processes: the decomposition of a bTSP, the division of an ant colony and the initialization of aggregated pheromone trails and aggregated heuristic matrices. Their detailed descriptions are as follows:

(a) *Decompose bTSP* This process uses the weight vectors to transform an original bTSP into a number of scalar subproblems problems and N aggregated distance matrices are generated, one for every subproblem. MoACO/D tackles the bTSP by solving these subproblems during the run. Given N evenly spread weight vectors $\{\boldsymbol{\lambda}^1, \boldsymbol{\lambda}^2, \ldots, \boldsymbol{\lambda}^N\}, \boldsymbol{\lambda}^k = (\lambda_1^k, \lambda_2^k)$, satisfying λ_1^k, $\lambda_2^k \ge 0$ and $\lambda_1^k + \lambda_2^k = 1$, $k = 1, \ldots, N$, a bTSP is converted into N scalar optimization subproblems using Tchebycheff approach. The objective function of each subproblem is formulated in (2). However, as usual, the process of finding the exact reference point $\boldsymbol{z}^*$ in (2) is very time consuming; hence, we use $\boldsymbol{z}$ for substitution at the initial step, i.e., $\boldsymbol{z} = [f_1(\pi), f_2(\pi)]$, $t = 1$, where $[f_1(\pi), f_2(\pi)]$ is the objective vector produced by a random solution π. Then along with the algorithm running, $\boldsymbol{z}$ will be updated.

(b) *Divide ant colony* To adapt the decomposition, a colony with N ants is divided into N subcolonies in an overlapped manner, which is similar to the proposal of Iredi et al. (2001). To be specific, the kth subcolony for the kth subproblem is denoted as $S(k), S(k) = \{i_1, i_2, \ldots, i_T\}$ and $1 < T < N$, where $i_1, i_2, \ldots, i_T$ are the indexes of the closest T weight vectors to $\boldsymbol{\lambda}^k$. Here, the Euclidean

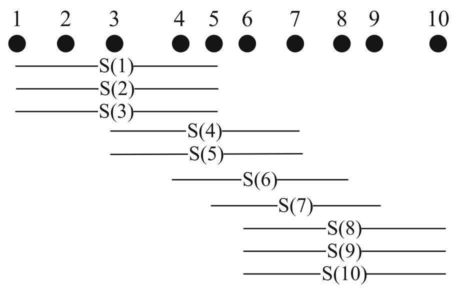


Fig. 2 An example for the division of an ant colony

distance is used to measure the closeness between any two weight vectors; thus, the closest weight vector to $\boldsymbol{\lambda}^k$ is itself, i.e., $i_1 = k$. Then apply $A(h), A(h) = \underset{k=1,2,\ldots,N}{\cup}\{k|h \in S(k)\}$, to represent a set of the indexes of the subproblems solved by the hth ant. The allocation leads to that ants belong to multiple subcolonies at the same time. Hence, each ant will solve at least one subproblem and construct more than one solution per iteration, that is, $|A(h)| \geq 1$. Besides, a relationship among subproblems is built by the use of $A(h)$, which will help to implement information share among subproblems during the run, as stated in the next step.

To clearly illustrate the division procedure, Fig. 2 shows an example with ten ants and five weight vectors in the neighborhood of each weight vector, i.e., $N = 10$ and $T = 5$. Therefore, in every iteration 50 solutions are constructed. In this figure, the ten ants are labeled as 1, 2, …, 10, denoted by heavy points, and the ten straight lines represent ten subproblems. The distances between each pair of adjacent ants are 2, 2, 3, 1, 1, 2, 2, 1, 3, respectively. A straight line under several ants implies the subproblems solved by these ants. In this division, $S(1) = S(2) = S(3) = \{1,2,3,4,5\}, S(4) = S(5) = \{3,4,5,6,7\}, S(6) = \{4,5,6,7,8\}, S(7) = \{5,6,7,8,9\}, S(8) = S(9) = S(10) = \{6,7,8,9,10\}, A(1) = A(2) = \{1,2,3\}, A(3) = \{1,2,3,4,5\}, A(4) = \{1,2,3,4,5,6\}, A(5) = \{1,2,3,4,5,6,7\}, A(6) = A(7) = \{4,5,6,7,8,9,10\}, A(8) = \{6,7,8,9,10\}, A(9) = \{7,8,9,10\}, A(10) = \{8,9,10\}$.

(c) *Initialize pheromone trails and heuristic matrices* In MoACO/D, each subproblem maintains an aggregated pheromone trail and an aggregated heuristic matrix, which are initialized using weight vector $\boldsymbol{\lambda}$ before iteration and will never be aggregated again during the run. For the kth subproblem, the pheromone trail $\mathbf{T}^k = [\tau_{ij}^k]_{L\times L}$ is initialized as

$$\tau_{ij}^k = \tau_0^k, \tag{3}$$

where τ_0^k is the initial pheromone trail whose value depends on the single-objective ACO optimizer used in MoACO/D. The heuristic matrix $\mathbf{I}^k = [\eta_{ij}^k]_{L\times L}$ is calculated by weighted cost values as

$$\eta_{ij}^k = 1 \Big/ \sum_{m=1}^{2} \lambda_m^k c_{ij}^m, \tag{4}$$

where c_{ij}^m is the cost value connecting the vertices i and j for the mth objective.

(2) Five processes are involved in this step: tour construction, pheromone trails update, reference point update, pheromone trail share and external archive update. Each process is further described as follows:

(a) *Construct tours* For every subcolony, each ant employs an ACO optimizer to construct a tour for corresponding subproblem using aggregated pheromone trail and heuristic matrix.

(b) *Update pheromone trails* After all ants have constructed their tours for corresponding subproblems, the pheromone trails are updated. This step depends on the ACO optimizer used in MoACO/D.

(c) *Update reference point* After all ants in $S(k)$ finish their construction of tours, each component z_m in reference point z is updated with the minimum of the objective value in $\{f_m(\pi^h)|h \in S(k)\}$, $m = 1, 2$, that is

$$z_m = \min_{h\in S(k)}\{f_m(\pi^h)\}. \tag{5}$$

Through update, the reference point z will approach the point z^*. Therefore, according to the idea of Tchebycheff decomposition, the solution of (2) will more likely be a Pareto solution of (1).

(d) *Share pheromone trails* Pheromone trail share is realized by the use of best ant b_k and the indexes set $A(b_k)$. First, the best ant b_k in $S(k)$ is the one that has a tour with the minimal fitness, i.e.,

$$b_k = \arg\min_{h\in S(k)}\{g_k(\pi^h|\boldsymbol{\lambda}^k, z)\}. \tag{6}$$

Then the tour π^{b_k} traveled by the ant b_k is applied to update the pheromone trails $\mathbf{T}^{p_1}, \mathbf{T}^{p_2}, \ldots, \mathbf{T}^{p_{|A(b_k)|}}$ of all the subproblems $p_1, p_2, \ldots, p_{|A(b_k)|}$ $(p_r \in A(b_k), r = 1, 2, \ldots, |A(b_k)|)$ solved by the ant b_k. To be specific, for the subproblem $p_r, p_r \in A(b_k)$, the pheromone levels $\tau_{ij}^{p_r}$ of all the arcs (i, j) belonging to the tour π^{b_k} is updated by

$$\tau_{ij}^{p_r} = \tau_{ij}^{p_r} + \Delta\tau^{p_r}, \quad (7)$$

where $\Delta\tau^{p_r}$ is the pheromone quantity deposited on the tour π^{b_k} for the subproblem p_r and is calculated by

$$\Delta\tau^{p_r} = 1 \Bigg/ \sum_{m=1}^{2} \lambda_m^{p_r} f_m(\pi^{b_k}). \quad (8)$$

The solution π^{b_k} is used to update the pheromone trails of more than one colony per iteration, i.e., the pheromone trails $\mathbf{T}^{p_r}$ of the subproblem $p_r, p_r \in A(b_k)$, $r = 1, 2, \ldots, |A(b_k)|$. The aim is to implement information share among subproblems solved by common ants, which will guide the search directions of the ants for other subproblems in the subsequent iterations.

(e) *Update archive* An external unbounded archive is used to collect Pareto solutions found during the run. To be specific, for the hth objective vector $F(\pi^h), h \in S(k)$, all the objective vectors dominated by $F(\pi^h)$ are removed from the archive, and if no objective vector in the archive dominates $F(\pi^h)$, $F(\pi^h)$ is added into the archive. When the algorithm stops, we obtain the final approximate Pareto solutions in the archive.

MoACO/D introduces the division strategy of an ant colony to specify the decomposition approach of a bTSP and uses a single-objective ACO optimizer to solve each subproblem. Thus, we can obtain a good Pareto front with a good approximation and an even distribution.

3.3 Characteristics

MoACO/D possesses some characteristics, i.e., (1) weights are assigned to subproblems rather than ants; (2) an ant belongs to multiple subcolonies at the same time and hence constructs more than one solution per iteration; and (3) the same solution π^{b_k} is further used for updating the pheromone trails of more than one subproblems to implement information share among subproblems. In the light of the taxonomy proposed by López-Ibáñez and Stützle (2010), MoACO/D belongs to the category that uses one pheromone trail and one heuristic matrix per colony, weighted sum aggregation and many weight vectors.

The first distinction between MoACO/D and the existing approaches is the way that handles multiple objectives during the run. In MoACO/D, the original MOP is transformed into a number of scalar optimization problems, whereas most existing approaches use a lexicographic ordering or cooperative search manner and the non-dominated sorting to handle the multiple objectives.

The other distinction is the pheromone trail update, including two aspects: which solution is utilized to update the pheromone trails and which pheromone trails are updated. Except for the general pheromone trails update as in single-objective ACO algorithms, MoACO/D employs a pheromone trail share procedure, which is also an pheromone trail update procedure. In MoACO/D, for each subproblem the optimal solution searched by an ant is used to update the pheromone trails of all the subproblems solved by the ant, which implements the information share among the subproblems, and hence guides the search direction in subsequent iterations, whereas in Iredi et al. (2001), two methods called update by origin and update by region are used. The former relates to that an ant only updates the pheromone trails in its own colony, and the latter is that after splitting the non-dominated front found so far into many parts with equal size, the ants found solutions in the ith part only update the pheromone trail of subcolony i. In López-Ibáñez et al. (2004), another pheromone update strategy is introduced. The non-dominated solutions are first distributed among colonies and then only the best solutions with respect to each objective are allowed to deposit pheromone in the respective pheromone trail of each colony.

4 MoACO/D variants

In this section, we present three MoACO algorithms by respectively combining MoACO/D with ant systems (AS) (Dorigo et al. 1996), Max–Min ant system (MMAS) (Stützle and Hoos 2000) and ant colony system (ACS) (Dorigo and Gambardella 1997). The three procedures in MoACO/D, pheromone trail initialization, tour construction and pheromone update, are specified. For simplicity, we name the three algorithms as MoACO/D-AS, MoACO/D-MMAS and MoACO/D-ACS, respectively.

4.1 MoACO/D-AS

AS is the first ACO algorithm developed by Dorigo et al. (1996). In AS, a good choice is to set the initial pheromone trails to N/C, where N is the number of ants and C is the length of a feasible tour. To make AS suitable for MoACO/D, for the kth subproblem the initial pheromone trail τ_0^k in (3) is assigned as

$$\tau_0^k = T \Bigg/ \sum_{m=1}^{2} \lambda_m^k f_m(\pi), \quad (9)$$

where π is a feasible tour of bTSP. Then for kth subproblem, each ant $h, h \in S(k)$, chooses a random vertex as its starting point, and it passes through the rest $L - 1$ vertices to form a whole tour. At each pace, the ant chooses the next unvisited vertex j when it stays at the vertex i following the probability

$$p(j) = \begin{cases} \frac{[\tau_{ij}^k]^\alpha [\eta_{ij}^k]^\beta}{\sum_{l \in \Omega(j)} [\tau_{il}^k]^\alpha [\eta_{il}^k]^\beta}, & \text{if } j \in \Omega(i) \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

where α and β are two parameters which determine the relative influence of the pheromone trail and the heuristic information, and $\Omega(i)$ is the set of vertices that the ant h has not visited yet. After all the ants have constructed their tours, the pheromone trail associated with every edge is evaporated by reducing all pheromones by a constant scale as

$$\tau_{ij}^k = (1-\rho)\tau_{ij}^k, \tag{11}$$

where $\rho \in (0,1]$ is the evaporation rate. Subsequently, all ants in $S(k)$ deposit pheromone on the arcs they have crossed in their tour as

$$\tau_{ij}^k = \tau_{ij}^k + \sum_{h \in S(k)} \Delta\tau_{ij}^h, \tag{12}$$

where $\Delta\tau_{ij}^h$ is the amount of pheromone that the ant h deposits on the arc (i, j), calculated as

$$\Delta\tau_{ij}^h = \begin{cases} 1/\sum_{m=1}^{2} \lambda_m^k f_m(\pi^h), & \text{if arc } (i,j) \in \pi^h \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

4.2 MoACO/D-MMAS

As compared with AS, MMAS introduces several modifications: pheromone trail limits, pheromone initialization and pheromone update. In MMAS, the recommended pheromone trail limits are $\tau_{\min} = \tau_{\max}(1-\sqrt[L]{0.05})/((L/2-1)\sqrt[L]{0.05})$ and $\tau_{\max} = 1/\rho C$, where C is the length of the optimal tour. The initial pheromone trail τ_0 is set to the upper pheromone trail $\tau_{\max}$. In MoACO/D-MMAS, we modify these characteristics to suit for bTSPs. For the kth subproblem, the upper pheromone trail $\tau_{\max}^k$ is initialized as

$$\tau_{max}^k = 1/\rho \sum_{m=1}^{2} \lambda_m^k f_m(\pi), \tag{14}$$

and once a new best tour is found, which is measured by the corresponding scalar objective function of kth subproblem, the value of $\tau_{\max}^k$ is updated. After all ants in $S(k)$ finish their tour construction, pheromone trails are updated by applying evaporation as in (3), followed by the deposit of new pheromone as

$$\tau_{ij}^k = (1-\rho)\tau_{ij}^k + \rho\Delta\tau^{b_k}, \forall(i,j) \in \pi^{b_k}, \tag{15}$$

where $\Delta\tau^{b_k} = 1/\sum_{m=1}^{2} \lambda_m^k f_m(\pi^{b_k})$ and b_k is the best ant determined by (6).

4.3 MoACO/D-ACS

ACS is another successor of AS and it introduces three major modifications, including modifying the transition rule, introducing local pheromone update and global pheromone update. In ACS, a recommended value for initializing the pheromone trails is set to $1/LC$, where C is the length of a feasible tour. To adapt this initialization to bTSP, for the kth subproblem, we initialize τ_0^k in (3) as

$$\tau_0^k = 1/L \sum_{m=1}^{2} \lambda_m^k f_m(\pi), \tag{16}$$

where π is a feasible tour of bTSP. It is worth noting that τ_0^k will be reinitialized after all ants in $S(k)$ have constructed their tours as done in MACS (Barán and Schaerer 2003). Then for kth subproblem, every ant in $S(k)$ chooses a random vertex as its starting point and chooses the next unvisited vertex j when it stays at the vertex i using a pseudorandom proportional transition rule as

$$j = \begin{cases} \arg\max_{l \in \Omega(i)} \{[\tau_{il}^k]^\alpha [\eta_{il}^k]^\beta\}, & \text{if } q \le q_0 \\ \hat{i}, & \text{otherwise,} \end{cases} \tag{17}$$

where q is a random number ranged in [0, 1]; q_0 is a constant between 0 and 1; $\hat{i}$ is a random variable selected by the probability in (10). Once an ant crosses the arc (i, j) the local pheromone update is performed using the formula

$$\tau_{ij}^k = (1-\xi)\tau_{ij}^k + \xi\tau_0^k, \tag{18}$$

where ξ is a preset constant varying in [0, 1]. After all ants have constructed their tours, the best ant is allowed to add pheromone in the tour π^{b_k}. In MoACO/D-ACS, this update procedure is implemented by (15). Additionally, the initial pheromone trail τ_0^k is reinitialized by applying the weighted average objective function values, i.e.,

$$\tau_0^k = 1 \Bigg/ \sum_{m=1}^{2} \lambda_m^k \hat{f}_m^k, \tag{19}$$

where $\hat{f}_m^k$ is the average of the mth objective value $f(\pi^h), h \in S(k), m = 1, 2$, that is,

$$\hat{f}_m^k = \sum_{h \in S(k)} f_m(\pi^h)/T. \tag{20}$$

5 Experimental results and analysis

In this section, we start from the construction of bTSP instances using benchmark TSPs and then turn to make appropriate configurations for three MoACO/D variants in an empirical way. Subsequently, extensive experiments are conducted on bTSPs. Experimental results compared with three existing MoACO approaches verify the efficiency and effectiveness of the presented MoACO/D framework.

5.1 bTSP instances

In this section, 8 bTSPs, called KroAB50, KroCD50, KroAB100, KroAD100, KroBC100, KroCD100, KroAB150 and KroAB200, having 50, 50, 100, 100, 100, 100, 150 and 200 cities, respectively, are constructed using pairs of benchmark TSPs and the same way as in García-Martínez et al. (2007). The benchmark TSPs, labeled KroA100, KroB100, KroC100, KroD100, kroA150, kroB150, KroA200 and KroB200, are available on the website http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/. To be specific, KroAB100, KroAD100, KroBC100, KroCD100, KroAB150 and KroAB200 are constructed using KroA100 and KroB100, KroA100 and KroD100, KroB100 and KroC100, KroC100 and KroD100, KroA150 and KroB150, KroA200 and KroB200, respectively. KroAB50 and KroCD50 are constructed using the first 50 cities of KroA100 and KroB100, and the first 50 cities of KroC100 and KroD100, respectively. All the benchmark TSPs used are completely independent; hence, the objectives of these bTSP instances are also independent and non-correlated. Furthermore, in order to show the scalability of the algorithm (how does the growth of the number of cities deteriorate the algorithm performance?), two extra instances with 300 cities, EucildAB300 and EucildCD300 downloaded from the website http://eden.dei.uc.pt/paquete/tsp/, are employed to construct the test bed. Due to difficulties and complexities of these bTSP instances, they are challenging enough to evaluate various multi-objective optimization algorithms (García-Martínez et al. 2007; Jaszkiewicz and Zielniewicz 2009; López-Ibáñez and Stützle 2010b).

5.2 Parameter setting

In MoACO/D, except for general parameters in ACO, another two important parameters are involved, the number N of subproblems or subcolonies or ants in colony and the number T of ants in each subcolony. To obtain appropriate configurations of three MoACO/D variants for further experiments, we use KroAB50 as an example to investigate the effects of these parameters on the performance of MoACO/D-AS, MoACO/D-MMAS and MoACO/D-ACS in an empirical way. In our experiments, the values 5, 10, 15, 20, 25, 30 for N and the values 2, 5, 10 for T are considered. Because of the condition $2 < T < N$ must be satisfied, there are 15 combinations of N and T. The other parameters are set to the recommended values in single-objective ACO algorithms (Dorigo and Stützle 2004): $\alpha = 1, \beta = 2$ for all MoACO/D variants, $\rho = 0.5, 0.02$ and 0.1 for MoACO/D-AS, MoACO/D-MMAS and MoACO/D-ACS, respectively, $q_0 = 0.9$ and $\xi = 0.1$ for MoACO/D-ACS. The weighted vectors are generated using a statistical method, that is, $\boldsymbol{\lambda}^k = (\lambda^k, 1 - \lambda^k)$, $k = 1, 2, \ldots, N$ and λ^k is a random variable following a uniform distribution in the range of (0, 1). The maximal number 2e4 of function evaluations (FEs) is used as the termination condition of all tests and each case is performed for 15 runs independently. All the algorithms in this contribution are implemented on the platform MATLAB 7.6a and all the tests are conducted on PC with 2.4 GHz CPU, 2 GB RAM and Windows XP OS.

Since the quantitative performance evaluation of a multi-objective optimizer is also an MOP (Zitzler et al. 2003), no single metric is able to reflect the performance of a multi-objective algorithm at every aspect. In this paper, several metrics are used: cardinality $|\mathcal{P}|$ of the approximation set $\mathcal{P}$, S metric, $R1_R$ metric, $R3_R$ metric and ϵ indicator (including $I_\epsilon(\mathcal{P}_0, \mathcal{P})$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0)$) (see Appendix). The S metric allows us to measure the diversity of approximation $\mathcal{P}$ and the proximity of $\mathcal{P}$ to the reference set $\mathcal{P}_0$. The $R1_R$ and $R3_R$ metrics measure the quality of $\mathcal{P}$ based on the reference set $\mathcal{P}_0$. The other two indicators, $I_\epsilon(\mathcal{P}_0, \mathcal{P})$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0)$ measure the extent of superiority of reference set $\mathcal{P}_0$ over $\mathcal{P}$. According to the definitions of these metrics, the best values of $S(\leq 1), R1_R(\geq 0.5)$, $R3_R(\leq 0), I_\epsilon(\mathcal{P}_0, \mathcal{P})(\leq 1)$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0)(\geq 1)$ are 1, 0.5, 0, 1 and 1, respectively.

In the experiments, for every MoACO/D variant, the reference set $\mathcal{P}_0$ is generated by the union of all solutions obtained from all runs and all cases except for the dominated ones. Tables 1, 2 and 3 provide the mean results of the metrics when T and N cover all combinations for MoACO/D-AS, MoACO/D-MMAS and MoACO/D-ACS, respectively. According to Tables 1 and 3, T and N could be assigned as 5 and 30 for both MoACO/D-AS and MoACO/D-ACS because they get the best results in 4 out of 6 metrics. Moreover, when T is fixed and as the number of N increases, the values of S and $|\mathcal{P}|$ increase. Also $R3_R$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0)$ obtain better results when N is a larger value. Table 2 manifests that $T = 2$ and $N = 20$ is an appropriate configuration for MoACO/D-MMAS since it obtains better overall results compared with other combinations. In the next section, these configurations are utilized to make comparisons with other three MoACO algorithms.

5.3 Comparisons with three MoACO approaches

The survey paper García-Martínez et al. (2007) made a convincingly comparative analysis of eight MoACO algorithms reported in literature and two well-known

Table 1 Mean values for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0)$) on KroAB50 of MoACO/D-AS with various T and N

Combinations	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$
$T = 2, N = 5$	32.8000	0.8605	**0.9895**	−0.5869	0.9937	1.4805
$T = 2, N = 10$	37.0000	0.8860	0.9965	−0.4204	0.9901	1.3344
$T = 2, N = 15$	35.7333	0.9034	0.9982	−0.3366	0.9942	1.2809
$T = 2, N = 20$	37.6000	0.9118	0.9982	−0.2703	0.9887	1.2174
$T = 2, N = 25$	38.0000	0.9077	0.9982	−0.2840	0.9911	1.2251
$T = 2, N = 30$	36.9333	0.9150	0.9982	−0.2537	0.9873	1.2061
$T = 5, N = 10$	34.8667	0.8856	0.9965	−0.4391	0.9911	1.3704
$T = 5, N = 15$	36.0667	0.8973	0.9965	−0.3870	0.9907	1.3528
$T = 5, N = 20$	39.1333	0.9148	0.9965	−0.2663	0.9926	1.2481
$T = 5, N = 25$	38.2000	0.9078	0.9965	−0.2940	**0.9952**	1.2707
$T = 5, N = 30$	**43.0667**	**0.9212**	0.9982	**−0.2299**	0.9946	**1.2045**
$T = 10, N = 15$	42.7333	0.9200	1.0000	−0.2599	0.9856	1.2383
$T = 10, N = 20$	36.5333	0.9042	1.0000	−0.3407	0.9872	1.3211
$T = 10, N = 25$	39.8000	0.9102	1.0000	−0.2915	0.9886	1.2564
$T = 10, N = 30$	37.8000	0.9106	1.0000	−0.2749	0.9875	1.2425

Table 2 Mean values for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0)$) on KroAB50 of MoACO/D-MMAS with various T and N

Combinations	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$
$T = 2, N = 5$	40.6667	0.9001	1.0000	−0.3825	0.9976	1.3925
$T = 2, N = 10$	43.6667	0.9286	0.9860	−0.2305	0.9996	1.2806
$T = 2, N = 15$	46.5333	0.9395	0.9982	−0.1699	0.9963	1.2008
$T = 2, N = 20$	**50.0000**	**0.9475**	**0.9912**	**−0.1397**	0.9972	1.1587
$T = 2, N = 25$	46.0000	0.9452	0.9982	−0.1429	0.9983	1.1750
$T = 2, N = 30$	49.9333	0.9493	**0.9912**	−0.1308	**0.9995**	**1.1435**
$T = 5, N = 10$	40.7333	0.9075	1.0000	−0.3108	0.9762	1.3041
$T = 5, N = 15$	46.4000	0.9153	1.0000	−0.2660	0.9832	1.2654
$T = 5, N = 20$	43.1333	0.9144	1.0000	−0.2646	0.9769	1.2422
$T = 5, N = 25$	49.9333	0.9225	1.0000	−0.2228	0.9750	1.2036
$T = 5, N = 30$	46.8667	0.9214	1.0000	−0.2266	0.9776	1.1978
$T = 10, N = 15$	44.1333	0.8954	1.0000	−0.3393	0.9563	1.2584
$T = 10, N = 20$	42.2667	0.8904	1.0000	−0.3755	0.9602	1.2953
$T = 10, N = 25$	46.4000	0.9000	1.0000	−0.3209	0.9598	1.2429
$T = 10, N = 30$	48.4667	0.9079	1.0000	−0.2738	0.9621	1.1985

evolutionay multi-objective optimization algorithms, NSGA-II (Deb et al. 2002) and SPEA2 (Zitzler et al. 2001), by using six bTSPs. The experimental results show that the three algorithms, MACS (Barán and Schaerer 2003), BIANT (Iredi et al. 2001; García-Martínez et al. 2007) and UNBI (Iredi et al. 2001; García-Martínez et al. 2007), outperform the other seven approaches. Therefore, we consider the three algorithms, MACS, BIANT and UNBI, as benchmark algorithms in the following experiments. To show the main characteristics of the three algorithms, we give a very brief review of them as follows:

- MACS (Barán and Schaerer 2003) is a variation of MACS-VRPTW (Gambardella et al. 1999) proposed by Gambardella et al. Contrary to its predecessor, MACS uses a single pheromone trail and several heuristic matrices. During the local pheromone update, the amount of pheromone deposited on the tour is not fixed, but will be reinitialized at the end of each iteration using the average values of Pareto solutions. The global pheromone update is performed using every solution of the current approximation set after an iteration.

Table 3 Mean values for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0,\mathcal{P}), I_\epsilon(\mathcal{P},\mathcal{P}_0)$) on KroAB50 of MoACO/D-ACS with various T and N

Combinations	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0,\mathcal{P})$	$I_\epsilon(\mathcal{P},\mathcal{P}_0)$
$T = 2, N = 5$	38.7333	0.8782	1.0000	−0.5099	0.9926	1.4548
$T = 2, N = 10$	48.7333	0.9189	0.9982	−0.2631	0.9929	1.2804
$T = 2, N = 15$	57.2667	0.9347	1.0000	−0.1833	0.9930	1.2246
$T = 2, N = 20$	58.4000	0.9430	0.9982	−0.1501	0.9952	1.1734
$T = 2, N = 25$	57.2667	0.9469	1.0000	−0.1494	0.9946	1.1522
$T = 2, N = 30$	62.9333	0.9526	1.0000	−0.1283	0.9927	**1.1296**
$T = 5, N = 10$	51.8667	0.9206	0.9982	−0.2656	0.9972	1.2884
$T = 5, N = 15$	56.4667	0.9355	0.9982	−0.1892	0.9972	1.2347
$T = 5, N = 20$	61.6000	0.9413	0.9930	−0.1540	**0.9988**	1.2157
$T = 5, N = 25$	64.6667	0.9496	**0.9930**	−0.1192	0.9980	1.1731
$T = 5, N = 30$	**66.8667**	**0.9559**	**0.9930**	**−0.1022**	0.9985	1.1617
$T = 10, N = 15$	49.4667	0.9182	0.9947	−0.2413	0.9976	1.2814
$T = 10, N = 20$	55.5333	0.9344	0.9965	−0.1876	0.9965	1.2283
$T = 10, N = 25$	56.5333	0.9410	1.0000	−0.1524	0.9964	1.2044
$T = 10, N = 30$	60.1333	0.9431	1.0000	−0.1331	0.9976	1.1872

- BIANT (Iredi et al. 2001; García-Martínez et al. 2007) is characterized by multiple pheromone trails and heuristic matrices aggregated by weighted product. As it was originally designed for a bi-criteria vehicle routing problem, García-Martínez et al. modified it to suit for bTSP, i.e., each ant which generated non-dominated solutions at the current iteration is used to update pheromone trail, and the amount of pheromone deposited on the trip is the inverse of the cost of its solution according to the objective function.
- UNBI (Iredi et al. 2001; García-Martínez et al. 2007) is a modified version of BIANT, which divides an ant colony into Nc subcolonies and each objective in a subcolony has a pheromone trail. When pheromone trails are updated, each ant only contributes to those pheromone trails associated with the subcolony that it belongs to.

To compare the performance of these six algorithms, ten bTSPs with various cites discussed in Sect. 5.1 are considered. To make a fair comparison, we tuned the number N of ants of MACS, BIANT and UNBI and the number Nc of colony of UNBI in the same way as done in previous discussion of MoACO/D variants. Then we chose the best configurations for them to carry out experiments. All parameter values for the algorithms are presented in Table 4. We independently execute each algorithm on each bTSP instance for 15 runs; thus for every bTSP, 15 approximation sets can be obtained for each algorithm.

The presentation of all approximation sets of each algorithm in a single graph is usually confusing and misleading. It is difficult to separate out individual approximation sets and to clearly understand the distribution of the location and extent of the different approximation sets over multiple runs (Knowles 2005). Since an attainment surface can emphasize the distribution and indicate the quality of the individual point, and the visualization of attainment surface allows us to characterize and/or summarize the behavior of a single algorithm in many runs in a graphical manner, we use the attainment surface to visualize the outcomes of the experimental results.

Attainment surface, originally proposed in Fonseca and Fleming (1996), relates to a boundary which separates the objective space into two parts: the objective vectors that are attained by the outcomes returned by an optimizer and

Table 4 Parameter values considered

Parameters	Values
Number of runs	15
Number of FEs	2e4 (≤200 cities) and 3e4 (≥300cities)
N	20 for MoACO/D-MMAS and MACS
	30 for MoACO/D-AS and MoACO/D-ACS
	50 for UNBI
	60 for BIANT
ρ	0.1 for MACS, BIANT and UNBI (García-Martínez et al. 2007)
	0.5 for MoACO/D-AS
	0.02 for MoACO/D-MMAS
	0.1 for MoACO/D-ACS
T	2 for MoACO/D-MMAS
	5 for MoACO/D-AS and for MoACO/D-ACS
ξ	0.1 for MACS and MoACO/D-ACS
Nc	5 for UNBI
α	1
β	2
q_0	0.9 for MACS and MoACO/D-ACS

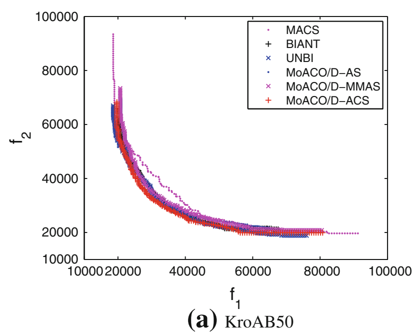
those that are not. The notion of attainment surface is formalized in the concept of %-attainment surface, which is the line separating the objective space attained by % of the runs of an algorithm (López-Ibáñez et al. 2010). In this contribution, the median attainment surface, which delimits the region attained by 50% of the runs, is considered to present the average behavior of MoACO/D as well as three benchmark algorithms. We use the software programmed by Knowles (2005), which can be downloaded from the website http://dbkgroup.org/knowles/plot_attainments/, to implement the function, and the resolution 100 is considered in the paper.

The experimental results obtained by the six algorithms are shown in Fig. 3, where each curve provides an estimation of the median attainment surface obtained from 15 approximation sets for a specific algorithm and a bTSP and it implies the location, dispersion, and even skewness of the distribution of solutions in each region of the trade-off surface. From the median attainment surfaces in every plot, three MoACO/D variants would appear to produce good results more often than MACS, BIANT and UNBI on large bTSPs. Specifically, three MoACO/D variants obtain similar median attainment surfaces on all cases and obtain better median attainment surfaces in the center region of the trade-off surface compared with MACS, BIANT and UNBI, while they obtain slightly worse median attainment surfaces in the upper-left and lower-right regions of the trade-off surface than MACS and UNBI. BIANT and UNBI perform similarly. Whereas, along with the increase of the number of cities, the results obtained by BIANT and UNBI are deteriorated severely in the central regions of the trade-off surface, especially in the case of EuclidAB300 and EuclidCD300. As for MACS, the median attainment surfaces are not as good as other algorithms in the center region of the trade-off surface on all bTSPs. However, MACS possesses best spreads and its performance does not degenerate so badly as that of BIANT and UNBI when large-scale bTSPs are considered. According to the plots, MoACO/D variants seem to best approximate to real Pareto front in most regions and MACS seems to generate approximation sets with best spreads. Moreover, we can infer from the plots that MoACO/D-ACS perform best in the three MoACO/D variants.

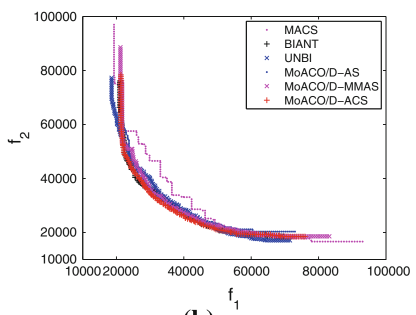
To quantitatively compare the performance of six algorithms, six metrics used in the section of parameter setting and the average computation time are considered. For each bTSP the reference set $\mathcal{P}_0$ is generated by the union of all solutions obtained from 6 algorithm and 15 runs, except for the dominated ones. Tables 5, 6, 7, 8, 9, 10, 11, 12, 13 and 14 summarize the statistics of these metrics on ten bTSPs. Through analyzing the statistical results, we can draw some conclusions:

- Except for KroAB50 and KroCD50, the numbers of non-dominated solutions obtained by MoACO/D-MMAS and MoACO/D-ACS are larger than other approaches, especially much larger than MACS. In the cases of KroAB50 and KroCD50, UNBI gets more non-dominated solutions than other algorithms and BIANT gets more solutions than UNBI on KroAB150, KroAB200, EuclidAB300 and EuclidCD300. As the cardinality of the approximation set could not reflect the quality of solutions, we cannot say that MoACO/D variants outperform others at this moment, and vice versa.
- Except for KroAB50 and KroCD50, MoACO/D-ACS obtains the best mean S values which affirms the approximation set obtained by MoACO/D-ACS has best diversity and best approximates to the pseudo Pareto set on large bTSPs. MoACO/D-MMAS is the second best approach on KroAB150, KroAB200, EuclidAB300 and EuclidCD300 while MoACO/D-AS performs better than MACS, BIANT and UNBI on EuclidAB300 and Euclid-CD300. Moreover, according to S values UNBI is the best approach on KroAB50 and KroCD50 and MACS is the worst one on all bTSPs.
- For all bTSPs, MoACO/D-ACS obtains the best mean values for metrics $R1_R$ and $R3_R$, that is, the values of $R1_R$ and $R3_R$ best approach to 0.5 and 0, respectively, among the six algorithms. The results indicate that MoACO/D-ACS has the largest probability to produce the approximation set well approximating to the pseudo Pareto set among six algorithms. In addition, according to the statistical values, MoACO/D-AS and MoACO/D-MMAS outperforms MACS, BIANT and UNBI on KroAB150, KroAB200, EuclidAB300 and Euclid-CD300 and the performance of MACS is poorer than other algorithms referring to $R1_R$ and $R3_R$ metrics.
- For all bTSPs, MoACO/D-ACS obtains the best mean values in terms of $I_\epsilon(\mathcal{P}_0, \mathcal{P})$ and obtains slight worse values than BIANT or UNBI with respect to $I_\epsilon(\mathcal{P}, \mathcal{P}_0)$, which implies that the extent of superiority of reference set $\mathcal{P}_0$ over sets generated by MoACO/D-ACS is less than other algorithms to some extent. As for MoACO/D-AS and MoACO/D-MMAS, medium values are obtained. We can also learn from the results that MACS gets the worst values, which implies that reference set $\mathcal{P}_0$ dominates the sets returned by MACS to a large extent.
- Referring to average computation time, three MoACO/D variants cost much less time than other algorithms, especially on larger bTSPs, which reflects the efficiency of MoACO/D framework. The reason is that calculating the probability of next vertex to be visited is much simpler in MoACO/D because it only uses one

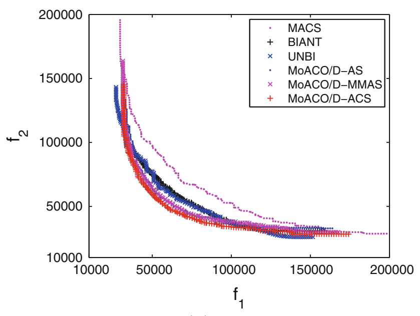
Fig. 3 Median attainment surfaces from 15 approximation sets



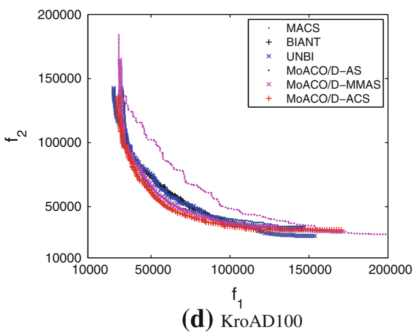
(a) KroAB50



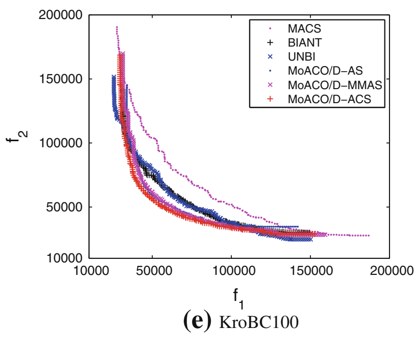
(b) KroCD50



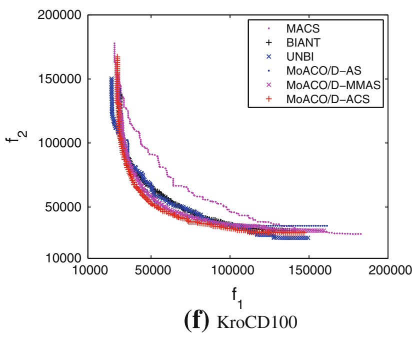
(c) KroAB100



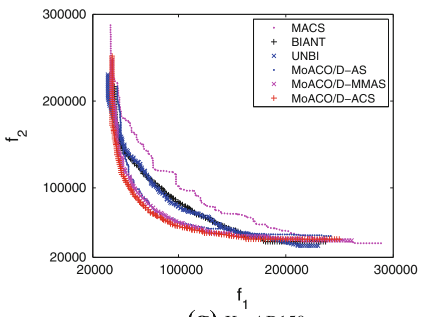
(d) KroAD100



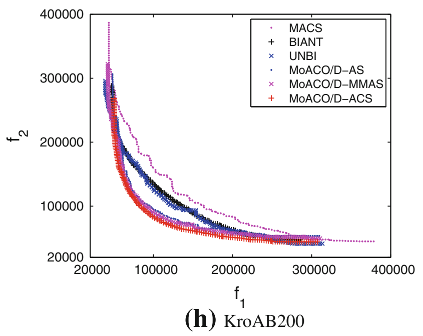
(e) KroBC100



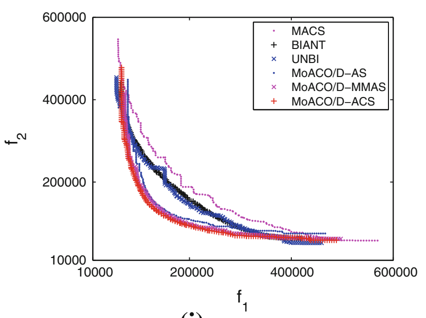
(f) KroCD100



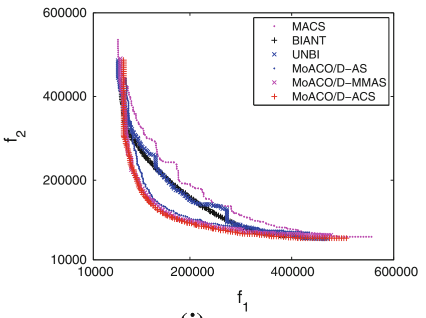
(g) KroAB150



(h) KroAB200



(i) EuclidAB300



(j) EuclidCD300

Table 5 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on KroAB50

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	43.2667	0.9055	1.0000	−0.3782	0.9581	1.2121	47.8649
	SD	12.2909	0.0293	0	0.1064	0.0284	0.0502	–
BIANT	Mean	49.5333	0.9429	1.0000	−0.1632	0.9925	1.1697	54.5607
	SD	5.8903	0.0054	0	0.0208	0.0059	0.0190	–
UNBI	Mean	**77.4000**	**0.9645**	0.9947	−0.1385	0.9992	**1.1318**	93.5855
	SD	14.6716	0.0061	0.0109	0.0324	0.0017	0.0345	–
MoACO/D-AS	Mean	40.4667	0.9231	0.9965	−0.2537	0.9946	1.2509	48.4737
	SD	5.8416	0.0110	0.0093	0.0824	0.0082	0.0907	–
MoACO/D-MMAS	Mean	47.5333	0.9203	1.0000	−0.2782	0.9749	1.2564	49.7298
	SD	5.8293	0.0151	0	0.0815	0.0123	0.0930	–
MoACO/D-ACS	Mean	66.8000	0.9585	**0.9754**	**−0.1040**	**0.9999**	1.1700	**45.9503**
	SD	6.9096	0.0129	0.0233	0.0426	0.0004	0.0689	–

Table 6 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on KroCD50

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	29.4000	0.8617	1.0000	−0.4780	0.9325	1.3160	48.1959
	SD	11.4754	0.0263	0	0.0864	0.0250	0.0535	–
BIANT	Mean	59.6667	0.9423	0.9965	−0.1488	0.9950	1.2034	53.9702
	SD	6.1489	0.0065	0.0136	0.0250	0.0068	0.0321	–
UNBI	Mean	**80.0667**	**0.9545**	0.9965	−0.1610	0.9978	**1.1558**	89.5520
	SD	8.2848	0.0081	0.0093	0.0243	0.0048	0.0419	–
MoACO/D-AS	Mean	44.2667	0.9160	0.9965	−0.2735	0.9962	1.3340	48.9076
	SD	5.8367	0.0109	0.0136	0.0685	0.0086	0.0824	–
MoACO/D-MMAS	Mean	54.7333	0.9140	1.0000	−0.2896	0.9791	1.3426	**47.0746**
	SD	8.6228	0.0251	0	0.1477	0.0075	0.1693	–
MoACO/D-ACS	Mean	71.2667	0.9489	**0.9719**	**−0.1248**	**1.0000**	1.2279	47.1364
	SD	7.0048	0.0147	0.0337	0.0621	0	0.0886	–

Table 7 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on KroAB100

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	20.4000	0.8603	1.0000	−0.6051	0.9035	1.4215	91.7660
	SD	1.8048	0.0070	0	0.0267	0.0253	0.0335	–
BIANT	Mean	55.4000	0.9255	1.0000	−0.2922	0.9771	**1.2535**	124.8941
	SD	6.5334	0.0062	0	0.0186	0.0160	0.0238	–
UNBI	Mean	65.2667	0.9433	0.9965	−0.2922	0.9949	1.3015	181.3638
	SD	11.3921	0.0079	0.0093	0.0339	0.0074	0.0669	–
MoACO/D-AS	Mean	51.9333	0.9244	0.9895	−0.2655	0.9906	1.3772	78.6134
	SD	7.2945	0.0083	0.0278	0.0480	0.0097	0.0965	–
MoACO/D-MMAS	Mean	68.2667	0.9297	1.0000	−0.2508	0.9797	1.3807	77.1714
	SD	9.7356	0.0144	0	0.0810	0.0106	0.1908	–
MoACO/D-ACS	Mean	**72.2000**	**0.9541**	**0.9807**	**−0.1146**	**0.9999**	1.3006	**75.7100**
	SD	7.3698	0.0072	0.0337	0.0206	0.0004	0.0654	–

Table 8 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0,\mathcal{P}), I_\epsilon(\mathcal{P},\mathcal{P}_0), Time$) on KroAD100

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0,\mathcal{P})$	$I_\epsilon(\mathcal{P},\mathcal{P}_0)$	Time (s)
MACS	Mean	20.6667	0.8454	1.0000	−0.5806	0.9197	1.4183	116.5159
	SD	1.9149	0.0075	0	0.0228	0.0376	0.0195	–
BIANT	Mean	55.2667	0.9236	0.9982	−0.2462	0.9767	1.2972	122.0378
	SD	8.3961	0.0044	0.0068	0.0168	0.0172	0.0310	–
UNBI	Mean	66.2000	0.9425	0.9965	−0.2333	0.9915	**1.2458**	181.7274
	SD	9.1042	0.0117	0.0093	0.0439	0.0113	0.0682	–
MoACO/D-AS	Mean	52.2667	0.9071	1.0000	−0.3120	0.9868	1.4991	89.8358
	SD	6.4749	0.0131	0	0.0670	0.0078	0.1170	–
MoACO/D-MMAS	Mean	70.4000	0.9278	1.0000	−0.2238	0.9786	1.3681	77.6268
	SD	10.8746	0.0083	0	0.0348	0.0126	0.0964	–
MoACO/D-ACS	Mean	**74.8667**	**0.9508**	**0.9719**	**−0.1084**	**1.0000**	1.3143	**76.7397**
	SD	7.5296	0.0106	0.0351	0.0472	0	0.1012	–

Table 9 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0,\mathcal{P}), I_\epsilon(\mathcal{P},\mathcal{P}_0), Time$) on KroBC100

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0,\mathcal{P})$	$I_\epsilon(\mathcal{P},\mathcal{P}_0)$	Time (s)
MACS	Mean	20.6000	0.8327	1.0000	−0.6252	0.8983	1.4806	90.8578
	SD	2.2297	0.0067	0	0.0271	0.0175	0.0371	–
BIANT	Mean	59.7333	0.9134	1.0000	−0.3041	0.9768	**1.3192**	122.4397
	SD	7.2454	0.0042	0	0.0191	0.0163	0.0220	–
UNBI	Mean	65.9333	0.9285	0.9982	−0.3084	0.9979	1.3294	182.2919
	SD	8.2416	0.0132	0.0068	0.0446	0.0055	0.0804	–
MoACO/D-AS	Mean	48.0000	0.9076	0.9982	−0.3112	0.9844	1.5510	79.0867
	SD	5.9761	0.0117	0.0068	0.0678	0.0132	0.1482	–
MoACO/D-MMAS	Mean	67.5333	0.9168	1.0000	−0.2811	0.9782	1.4913	77.9049
	SD	11.4447	0.0175	0	0.0995	0.0086	0.2197	–
MoACO/D-ACS	Mean	**70.7333**	**0.9457**	**0.9702**	**−0.1387**	**0.9997**	1.4032	**76.3431**
	SD	9.6026	0.0145	0.0420	0.0604	0.0010	0.1613	–

Table 10 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0,\mathcal{P}), I_\epsilon(\mathcal{P},\mathcal{P}_0), Time$) on KroCD100

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0,\mathcal{P})$	$I_\epsilon(\mathcal{P},\mathcal{P}_0)$	Time (s)
MACS	Mean	21.2000	0.8410	1.0000	−0.6118	0.9128	1.4398	91.7760
	SD	3.0048	0.0086	0	0.0297	0.0265	0.0284	–
BIANT	Mean	56.2000	0.9243	0.9982	−0.2554	0.9791	1.2998	121.8478
	SD	3.5295	0.0040	0.0068	0.0161	0.0156	0.0311	–
UNBI	Mean	68.2667	0.9435	0.9982	−0.2485	0.9955	**1.2707**	181.4508
	SD	10.1803	0.0089	0.0068	0.0422	0.0070	0.0765	–
MoACO/D-AS	Mean	47.1333	0.9085	1.0000	−0.3202	0.9825	1.4907	89.4001
	SD	6.7704	0.0169	0	0.1106	0.0151	0.1901	–
MoACO/D-MMAS	Mean	66.0667	0.9233	1.0000	−0.2473	0.9762	1.3894	76.8204
	SD	11.4235	0.0162	0	0.0784	0.0114	0.1460	–
MoACO/D-ACS	Mean	**73.9333**	**0.9505**	**0.9702**	**−0.1206**	**1.0000**	1.3334	**76.4027**
	SD	12.0147	0.0150	0.0312	0.0590	0	0.1332	–

Table 11 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on KroAB150

Algorithm	Statistics	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	22.9333	0.8481	1.0000	−0.6769	0.9067	1.5225	164.4978
	SD	2.3745	0.0051	0	0.0181	0.0246	0.0354	–
BIANT	Mean	69.8667	0.9090	0.9982	−0.3923	0.9527	**1.3144**	238.5683
	SD	7.9000	0.0033	0.0068	0.0096	0.0233	0.0129	–
UNBI	Mean	66.4667	0.9197	0.9982	−0.4103	0.9952	1.4163	336.1009
	SD	10.8947	0.0096	0.0068	0.0378	0.0067	0.0935	–
MoACO/D-AS	Mean	53.2000	0.9111	1.0000	−0.3328	0.9739	1.5749	121.9835
	SD	10.1644	0.0170	0	0.1060	0.0132	0.2244	–
MoACO/D-MMAS	Mean	**80.9333**	0.9437	0.9965	−0.1887	0.9819	1.3583	128.0342
	SD	9.0116	0.0091	0.0136	0.0359	0.0122	0.1165	–
MoACO/D-ACS	Mean	76.2000	**0.9505**	**0.9702**	**−0.1257**	**1.0000**	1.4273	**99.5643**
	SD	10.6918	0.0154	0.0196	0.0668	0	0.1460	–

Table 12 Statistical values (mean and standard deviation) for the metrics($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on KroAB200

Algorithm	Statistics	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	22.2667	0.8608	1.0000	−0.7049	0.9176	1.5225	257.4959
	SD	1.7915	0.0046	0	0.0224	0.0244	0.0301	–
BIANT	Mean	79.9333	0.9122	1.0000	−0.4430	0.9555	**1.3348**	392.8706
	SD	6.8083	0.0034	0	0.0156	0.0217	0.0102	–
UNBI	Mean	76.0667	0.9221	1.0000	−0.4576	0.9889	1.4467	542.7912
	SD	7.8328	0.0129	0	0.0647	0.0122	0.1485	–
MoACO/D-AS	Mean	57.5333	0.9213	1.0000	−0.3165	0.9671	1.5422	146.7924
	SD	8.9272	0.0111	0	0.0575	0.0108	0.1501	–
MoACO/D-MMAS	Mean	**94.6667**	0.9427	0.9965	−0.2270	0.9866	1.4959	**137.5741**
	SD	9.5892	0.0162	0.0136	0.0804	0.0138	0.2039	–
MoACO/D-ACS	Mean	88.7333	**0.9578**	**0.9667**	**−0.1230**	**1.0000**	1.4289	138.9646
	SD	12.5155	0.0127	0.0289	0.0579	0	0.1604	–

Table 13 Statistical values (mean and standard deviation) for the metrics($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on EuclidAB300

Algorithm	Statistics	$\lvert\mathcal{P}\rvert$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	23.8000	0.8388	1.0000	−0.7024	0.9094	1.6128	816.7802
	SD	2.0771	0.0067	0	0.0272	0.0311	0.0393	–
BIANT	Mean	107.1333	0.8983	1.0000	−0.4643	0.9618	**1.3996**	1,421.4256
	SD	10.1550	0.0020	0	0.0115	0.0169	0.0134	–
UNBI	Mean	84.8667	0.8902	1.0000	−0.5389	0.9963	1.6442	1,705.0013
	SD	15.6062	0.0208	0	0.0960	0.0043	0.2078	–
MoACO/D-AS	Mean	65.0667	0.9063	1.0000	−0.3533	0.9571	1.7839	440.0125
	SD	8.8759	0.0119	0	0.0699	0.0094	0.2167	–
MoACO/D-MMAS	Mean	**115.9333**	0.9406	1.0000	−0.2034	0.9876	1.5748	393.9141
	SD	16.4641	0.0150	0	0.0701	0.0180	0.2143	–
MoACO/D-ACS	Mean	108.7333	**0.9563**	**0.9667**	**−0.1164**	**0.9997**	1.5078	**382.5195**
	SD	11.1897	0.0138	0.0592	0.0553	0.0012	0.1637	–

Table 14 Statistical values (mean and standard deviation) for the metrics ($|\mathcal{P}|, S, R1_R, R3_R, I_\epsilon(\mathcal{P}_0, \mathcal{P}), I_\epsilon(\mathcal{P}, \mathcal{P}_0), Time$) on EuclidCD300

Algorithm	Statistics	$\|\mathcal{P}\|$	S	$R1_R$	$R3_R$	$I_\epsilon(\mathcal{P}_0, \mathcal{P})$	$I_\epsilon(\mathcal{P}, \mathcal{P}_0)$	Time (s)
MACS	Mean	23.8000	0.8401	1.0000	−0.7013	0.9243	1.5722	816.2084
	SD	1.0823	0.0066	0	0.0290	0.0303	0.0306	–
BIANT	Mean	102.3333	0.8981	1.0000	−0.4685	0.9639	**1.3900**	1,344.0256
	SD	6.7153	0.0021	0	0.0137	0.0177	0.0143	–
UNBI	Mean	83.5333	0.8936	1.0000	−0.5230	0.9844	1.5493	1,696.8253
	SD	17.0833	0.0167	0	0.0714	0.0146	0.1524	–
MoACO/D-AS	Mean	69.4667	0.9108	1.0000	−0.3435	0.9566	1.6600	443.3698
	SD	10.8750	0.0195	0	0.1230	0.0117	0.2926	–
MoACO/D-MMAS	Mean	**121.0667**	0.9448	1.0000	−0.1817	0.9886	1.4682	397.8784
	SD	15.4433	0.0139	0	0.0586	0.0087	0.1940	–
MoACO/D-ACS	Mean	104.1333	**0.9500**	**0.9667**	**−0.1470**	**1.0000**	1.5618	**365.2076**
	SD	14.3221	0.0157	0.0289	0.0718	0	0.2179	–

pheromone trail and one heuristic matrix, whereas in MACS, BIANT and UNBI multiple pheromone trails and multiple heuristic matrices are used, which results in a higher computation time. However, MoACO/D simultaneously maintains many pheromone trails and heuristic matrices; hence more memory is required.

To make a comparison for any pair of algorithms, C metric (see Appendix) is employed. The metric evaluates the performance of two algorithms by mapping from the ordered pair sets into a number in the interval [0, 1]. The box plots based on the C metric are shown in Fig. 4, where each rectangle contains ten box plots representing the distribution of the C values for a certain ordered pair of algorithms. From left to right, ten box plots relate to KroAB50, KroCD50, KroAB100, KroAD100, KroBC100, KroCD100, KroAB150, KroAB200, EuclidAB300 and EucidCD300, respectively. In each rectangle, the bottom, middle and top dash lines refer to scales 0, 0.5 and 1, respectively. Each rectangle, representing algorithms $\mathcal{P}_1$ and $\mathcal{P}_2$ associated with the corresponding row and column, gives the fraction of $\mathcal{P}_2$ dominated by $\mathcal{P}_1$, that is $C(\mathcal{P}_1, \mathcal{P}_2)$.

The box plots clearly show that for all bTSPs MoACO/D-ACS dominates more than half of the outcomes returned

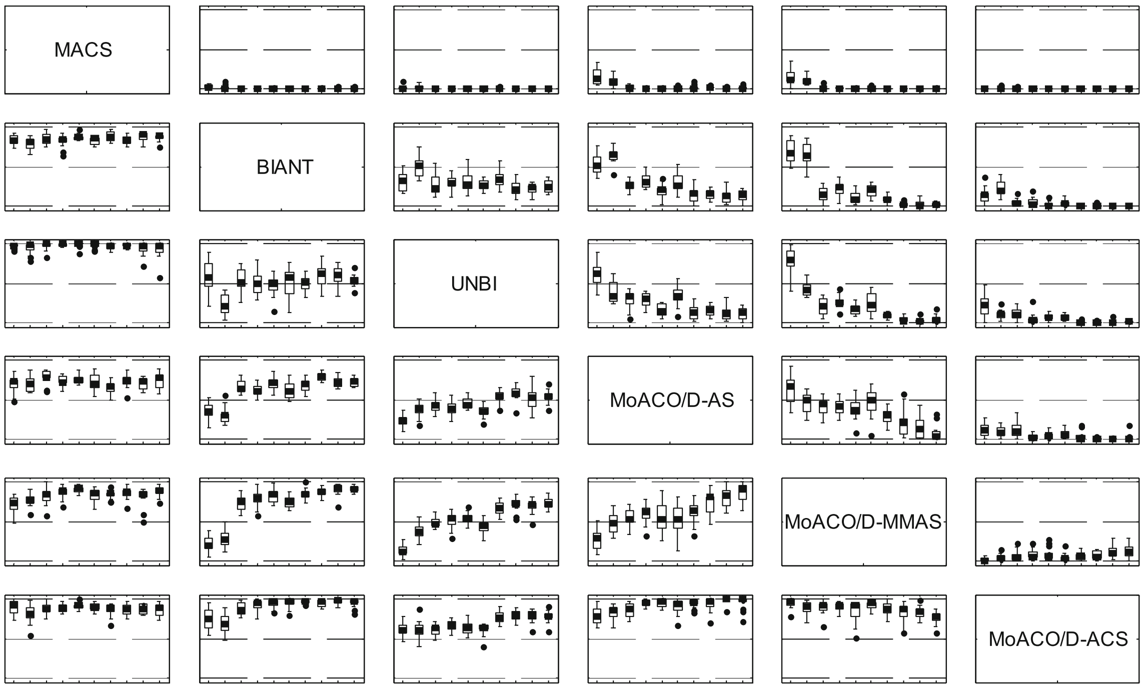


Fig. 4 Box plots based on the C metric

by MACS, BIANT and UNBI, while MACS has the worst performance among six algorithms with respect to C metric. The plots also illustrate that MoACO/D-AS and MoACO/D-MMAS perform better than BIANT on average over eight out of ten bTSPs, and MoACO/D-AS and MoACO/D-MMAS are superior to UNBI on large bTSPs. Moreover, the plots indicate that MoACO/D-ACS performs best among the three MoACO/D variants, followed by MoACO/D-MMAS.

Through extensive experiments and thorough analysis using many performance metrics as well as visualization of the median attainment surfaces, we can easily draw conclusions that MoACO/D framework is efficient and effective for bTSPs, especially on large cases, and the approach combining MoACO/D framework with ACS performs best against two other MoACO/D variants and three benchmark algorithms. The advantage of MoACO/D comes from the unique way that handles the MOP, i.e., MoACO/D transforms an original MOP into many scalar optimization subproblems using Tchebycheff approach and all subproblems are treated equally using single-objective ACO algorithm. Moreover, the pheromone trail share is designed to implement information share among subproblems, which also contributes to the performance of MoACO/D variants.

6 Conclusions

ACO is one of the most powerful population-based search approaches in solving a TSP, a well-known NP-hard combinatorial optimization problem. In the community of multiple objective optimization, how to obtain a Pareto front with good approximation and even distribution is always an ongoing issue. This paper proposes a framework, MoACO/D, for solving bTSPs based on Tchebycheff decomposition. In MoACO/D, a bTSP is decomposed into a number of scalar optimization subproblems and an ant colony is divided into a certain number of subcolonies with overlapping parts to suit for the decomposition. Three MoACO algorithms designed by, respectively, combining MoACO/D with AS, MMAS and ACS are presented. Extensive experiments performed on ten bTSPs with different complexities show that the MoACO/D framework is efficient and effective for solving bTSPs. Among the three MoACO/D variants, MoACO/D-ACS obtains better performance than three well-known MoACO algorithms and the other two variants are competitive to the benchmark MoACO algorithms on larger bTSPs in terms of several performance measures and median attainment surface. Our further work will focus on improving MoACO/D framework performance by introducing an appropriate local search technique and applying it to other multi-objective combinatorial problems.

Acknowledgments The authors are very grateful to the Area Editor, Prof. Carlos A. Coello Coello, and the anonymous reviewers for their insightful recommendations and comments to improve this paper. This work was supported by the National Natural Science Foundation of China (61170016), the Project Sponsored by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, the Scientific and Technological Funds for Young Scientists of Sichuan (09ZQ026-040), the Fund of Engineering Research Center for Transportation Safety, Ministry of Education (WHUTERCTS2010A01), by the Fundamental Research Funds for the Central Universities (SWJTU11ZT07, SWJTU09ZT10), and the Open Research Fund of Key Laboratory of Signal and Information Processing, Xihua University (SZJJ2009-003).

Appendix: Performance metrics

S metric

S metric, also named the hypervolume ratio (Zitzler and Thiele 1998), relates to the ratio of the hypervolume of an approximation set $\mathcal{P}$ and an optimum Pareto set or pseudo-optimum Pareto set $\mathcal{P}_0$, depicted in equation

$$S \triangleq = \frac{\mathrm{HV}(\mathcal{P})}{\mathrm{HV}(\mathcal{P}_0)}, \tag{21}$$

where $\mathrm{HV}(\mathcal{P})$ and $\mathrm{HV}(\mathcal{P}_0)$ are the hypervolumes of set $\mathcal{P}$ and $\mathcal{P}_0$, respectively, where the hypervolume relates to the area of coverage of a set and an anti-idea point with respect to the objective for a two-objective MOP, defined as

$$\mathrm{HV}(\mathcal{P}) \triangleq = \{\bigcup_i \mathrm{vol}_i | q_i \in \mathcal{P}\}, \tag{22}$$

where q_i is a non-dominated vector in $\mathcal{P}$; vol_i is the volume between the anti-idea solution and vector q_i. For a minimization MOP, it's assumed that the anti-idea point is the maximum value for each objective.

The S metric can be used as the basis of a dominance compliant comparison and possess the advantage of measuring both diversity and proximity. S value less than one indicates the approximation set $\mathcal{P}$ is not as good as $\mathcal{P}_0$ and if S value equals to 1, then $\mathcal{P} = \mathcal{P}_0$. Therefore, the larger the value S is, the better the $\mathcal{P}$ approximation set is.

$R1_R$ and $R3_R$ metrics

The $R1_R$ metric (Hansen 1998) calculates the probability that an reference set $\mathcal{P}_0$ is better than an approximation set $\mathcal{P}$ over a set of utility functions U, defined as

$$R1_R(\mathcal{P}, U, p) = \int_{u \in U} C(\mathcal{P}_0, \mathcal{P}, u) p(u)\, \mathrm{d}u, \tag{23}$$

where $u \in U$ is a utility function mapping each set to an utility measure; $p(u)$ is the probability density of the utility function u, and

$$C(\mathcal{P}_0, \mathcal{P}, u) = \begin{cases} 1 & : \text{if } u(\mathcal{P}_0) < u(\mathcal{P}) \\ 1/2 & : \text{if } u(\mathcal{P}_0) = u(\mathcal{P}) \\ 0 & : \text{if } u(\mathcal{P}_0) > u(\mathcal{P}) \end{cases} \quad (24)$$

with $u(\mathcal{P}) = \min_{q \in \mathcal{P}\{u(q)\}}$.

The $R3_R$ metric (Hansen and Jaszkiewicz 1998) measures the probability of superiority of an reference set $\mathcal{P}_0$ over approximation set $\mathcal{P}$, formulated as

$$R3_R(\mathcal{P}, U, p) = \int_{u \in U} \frac{u(\mathcal{P}_0) - u(\mathcal{P})}{u(\mathcal{P}_0)} p(u) \mathrm{d}u, \quad (25)$$

where U and $p(u)$ are defined as in $R1_R$ metric.

According to the metrics, the more the value $R1_R$ is near to $\frac{1}{2}$ or the smaller the value $R3_R$ is, the better the approximation set $\mathcal{P}$ is. Additionally, both $R1_R$ metric and $R3_R$ metric require a set of utility functions U which must be defined. In this contribution, the Tchebycheff utility function set (Hansen and Jaszkiewicz 1998) is used, that is,

$$\Bigg\{ u_\lambda(q, r) = \max_{j=1,2,\ldots,m} \{\lambda_j(q_j - r_j)\} | \lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m) \\ \cap \lambda_i \in \left\{ \frac{1}{k}, \frac{2}{k}, \ldots, \frac{k-1}{k} \right\} \cap \sum_{i=1}^{m} \lambda_i = 1 \Bigg\}. \quad (26)$$

Therefore, the original integration in (23) and (25) can be superseded by

$$R1(\mathcal{P}_1, \mathcal{P}_2, U, p) = \sum_{u_{\lambda,r} \in U} C(\mathcal{P}_1, \mathcal{P}_2, u_{\lambda,r}) p(u_{\lambda,r}) \quad (27)$$

$$R3(\mathcal{P}_1, \mathcal{P}_2, U, p) = \sum_{u_{\lambda,r \in U}} \frac{u_{\lambda,r}(\mathcal{P}_1) - u_{\lambda,r}(\mathcal{P}_2)}{u_{\lambda,r}(\mathcal{P}_1)} p(u_{\lambda,r}) \quad (28)$$

ϵ indicator

For a minimization problem with m positive objectives, an objective vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_m)$ is said to ϵ dominate another objective vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_m)$, written as $a \succcurlyeq_\epsilon b$, if and only if

$$\forall 1 \le i \le m : a_i \le \epsilon \cdot b_i \quad (29)$$

for a given $\epsilon > 0$. Given two approximation sets, $\mathcal{P}_1$ and $\mathcal{P}_2$, ϵ indicator measures (Zitzler et al. 2003) the smallest amount ϵ such that any solution $q_2 \in \mathcal{P}_2$ is ϵ dominated by at least one solution $q_1 \in \mathcal{P}_1$, i.e.,

$$I_\epsilon(\mathcal{P}_1, \mathcal{P}_2) = \min\{\epsilon | \forall b \in \mathcal{P}_2 \exists a \in \mathcal{P}_1 : a_\epsilon b\}. \quad (30)$$

When $I_\epsilon(\mathcal{P}_1, \mathcal{P}_2) < 1$, all solutions in $\mathcal{P}_2$ are dominated by a solution in $\mathcal{P}_1$. If $I_\epsilon(\mathcal{P}_1, \mathcal{P}_2) = 1$ and $I_\epsilon(\mathcal{P}_2, \mathcal{P}_1) = 1, \mathcal{P}_1$ and $\mathcal{P}_2$ represent the same approximation set. If $I_\epsilon(\mathcal{P}_1, \mathcal{P}_2) > 1$ and $I_\epsilon(\mathcal{P}_2, \mathcal{P}_1) > 1, \mathcal{P}_1$ and $\mathcal{P}_2$ are incomparable. When the optimum Pareto set or pseudo-optimum Pareto set is considered, i.e., $I_\epsilon(\mathcal{P}_0, \mathcal{P})$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0)$, $I_\epsilon(\mathcal{P}_0, \mathcal{P}) \le 1$ and $I_\epsilon(\mathcal{P}, \mathcal{P}_0) \ge 1$, and the more the value is near to 1, the better the set $\mathcal{P}$ is.

C metric

C metric (Zitzler and Thiele 1999) aims to evaluate the performance of two multi-objective algorithms by comparing the approximation Pareto sets. Let $\mathcal{P}_1, \mathcal{P}_2$ be two approximation Pareto sets obtained by two algorithms, the function C defines a mapping from the ordered pair $(\mathcal{P}_1, \mathcal{P}_2)$ to the interval [0, 1], i.e.,

$$C(\mathcal{P}_1, \mathcal{P}_2) = \frac{|\{\mathbf{b} \in \mathcal{P}_2 | \exists \mathbf{a} \in \mathcal{P}_1 : \mathbf{a} \succcurlyeq \mathbf{b}\}|}{|\mathcal{P}_2|}. \quad (31)$$

where $\mathbf{a} \succcurlyeq \mathbf{b}$ implies that the solution $\mathbf{a}$ dominates the solution $\mathbf{b}$. The value $C(\mathcal{P}_1, \mathcal{P}_2) = 1$ means that all objective vectors in $\mathcal{P}_2$ are dominated by at least one objective vector in $\mathcal{P}_1$. On the contrary, $C(\mathcal{P}_1, \mathcal{P}_2) = 0$ represents the case that no point in $\mathcal{P}_2$ is dominated by any point in $\mathcal{P}_1$. Note that both directions have to be considered, since $C(\mathcal{P}_1, \mathcal{P}_2)$ is not equal to $1 - C(\mathcal{P}_2, \mathcal{P}_1)$.

References

Angus D, Woodward C (2009) Multiple objective ant colony optimisation. Swarm Intell 3(1):69–85

Barán B, Schaerer M (2003) A multiobjective ant colony system for vehicle routing problem with time windows. In: Proceedings of the twenty first IASTED international conference on applied informatics, pp 97–102

Bullnheimer B, Hartl R, Strauss C (1999) An improved ant system algorithm for the vehicle routing problem. Ann Oper Res 89:319–328

Cardoso P, Jesus M, Márquez A (2011) ε-DANTE: an ant colony oriented depth search procedure. Soft Comput 15:149–182

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Doerner K, Gutjahr W, Hartl R, Strauss C, Stummer C (2006) Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection. Eur J Oper Res 171(3):830–841

Dorigo M, Gambardella L (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66

Dorigo M, Stützle T (2004) Ant colony optimization. MIT Press, Cambridge

Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Tran Syst Man Cybern Part B 26(1):29–41

Fonseca C, Fleming P (1996) On the performance assessment and comparison of stochastic multiobjective optimizers. In: Vogit H-M, Ebeling W, Rechenberg I, Schwefel HS (eds) Proceedings of PPSN-IV, fourth international conference on parallel problem solving from nature. Lecture notes in computer science, vol 1141. Springer, Berlin, pp 584–593

Gambardella L, Taillard E, Agazzi G (1999) MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In: Corne D, Dorigo M, Glover F (eds) New ideas in optimization. McGraw-Hill, pp 63–76

García-Martínez C, Cordón O, Herrera F (2007) A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. Eur J Oper Res 180(1): 116–148

Hansen M (1998) Metaheuristics for multiple objective combinatorial optimization. PhD thesis

Hansen M, Jaszkiewicz A (1998) Evaluating the quality of approximations to the non-dominated set. Tech. rep. Technical report IMM-REP-1998-7, Technical University of Denmark

Iredi S, Merkle D, Middendorf M (2001) Bi-criterion optimization with multi colony ant algorithms. In: Zitzler E, Deb K, Thiele L, Coello C, Corne D (eds) First International Conference on evolutionary multi-criterion optimization. Lecture notes in computer science, vol 1993. Springer, Berlin, pp 359–372

Ishibuchi H, Murata T (1998) A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Trans Syst Man Cybern Part C Appl Rev 28(3):392–403

Jaszkiewicz A (2002) On the performance of multiple-objective genetic local search on the 0/1 knapsack problem—a comparative experiment. IEEE Trans Evol Comput 6(4):402–412

Jaszkiewicz A, Zielniewicz P (2009) Pareto memetic algorithm with path relinking for bi-objective traveling salesperson problem. Eur J Oper Res 193:885–890

Knowles J (2005) A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In: Proceedings of the 5th international conference on intelligent systems design and applications. IEEE, Washington, DC, USA, pp 552–557

López-Ibáñez M, Stützle T (2010a) Automatic configuration of multiobjective ant colony optimization algorithms. In: Dorigo M, Birattari M, Di Caro G, Doursat R, Engelbrecht A, Floreano D, Gambardella L, Grob R, Sahin E, Stützle T, Sayama H (eds) ANTS 2010. Lecture notes in computer science, vol 6234, Springer, Berlin, pp 95–106

López-Ibáñez M, Stützle T (2010b) The impact of design choices of multiobjective ant colony optimization algorithms on performance: an experimental study on the biobjective TSP. In: Proceedings of the 12th annual conference on genetic and evolutionary computation, ACM, Portland, Oregon, USA, pp 71–78

López-Ibáñez M, Paquete L, Stützle T (2004) On the design of ACO for the biobjective quadratic assignment problem. In: Dorigo M (ed) Proceedings of ANTS. Lecture notes in computer science, vol 3172, Springer, Heidelberg, pp 224–225

López-Ibáñez M, Paquete L, Stützle T (2010) Empirical methods for the analysis of optimization algorithms. In: Dorigo M, Birattari M, Di Caro G, Doursat R, Engelbrecht A, Floreano D, Gambardella L, Grob R, Sahin E, Stützle T, Sayama H (eds) Exploratory analysis of stochastic local search algorithms in biobjective optimization. Springer, Berlin, pp 209–222

Mariano C, Morales E (1999) A multiple Ant-Q algorithm for the design of water distribution irrigation network. Tech. rep. Technical report HC-9904, Instituto Mexicano de Tecnología del Agua

Miettinen K (1999) Nonlinear multiobjective optimization. Kluwer, Boston

Peng W, Zhang Q, Li H (2009) Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem. In: Goh C, Ong Y, Tan K (eds) Multi-objective memetic algorithms. Studies in computational intelligence, vol 171. Springer, Berlin, pp 309–324

Stützle T, Hoos H (2000) MAX–MIN ant system. Future Gener Comput Syst 16(8):889–914

T'kindt V, Monmarché N, Tercinet F, Laügt D (2002) An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. Eur J Oper Res 142(2):250–257

Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731

Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Eiben A, Bäck T, Schaerer M, Schwefel H (eds) Parallel problem solving from nature V. Springer, Berlin, pp 292–301

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength pareto evolutionary algorithm. Tech. rep. Switzerland, Tech. rep. TIK-Rep, 103

Zitzler E, Thiele L, Laumanns M, Fonseca C, Fonseca V (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput 7(2):117–132