Computers and Mathematics with Applications 66 (2013) 1857–1868

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

A decomposition based estimation of distribution algorithm for multiobjective traveling salesman problems

Aimin Zhou, Feng Gao, Guixu Zhang *

Department of Computer Science and Technology, East China Normal University, 500 Dongchuan Road, Shanghai, China

ARTICLE INFO

Keywords:
Traveling salesman problem
Multiobjective optimization
MOEA/D

ABSTRACT

The *traveling salesman problem (TSP)* is a well known NP-hard benchmark problem for discrete optimization. However, there is a lack of TSP test instances for multiobjective optimization and some current TSP instances are combined to form a *multiobjective TSP (MOTSP)*. In this paper, we present a way to systematically design MOTSP instances based on current TSP test instances, of which the degree of conflict between the objectives is measurable. Furthermore, we propose an approach, named *multiobjective estimation of distribution algorithm based on decomposition (MEDA/D)*, which utilizes decomposition based techniques and probabilistic model based methods, to tackle the newly designed MOTSP test suite. In MEDA/D, an MOTSP is decomposed into a set of scalar objective sub-problems and a probabilistic model, using both priori and learned information, is built to guide the search for each sub-problem. By the cooperation of neighbor sub-problems, MEDA/D could optimize all the sub-problems simultaneously and thus find an approximation to the original MOTSP in a single run. The experimental results show that MEDA/D outperforms MOACO and MOEA/D-ACO, two ant colony based methods, on most of the given test instances and MEDA/D is insensible to its control parameters.

1. Introduction

The target of a *traveling salesman problem (TSP)* is to find the shortest tour for a salesman who would like to go from his home city to some other cities and come back to his home city, subject that each city will be visited once and only once. It has been proved that TSP is NP-hard [1]. Due to their simplicity to describe and hardness to solve, TSP instances have become combinational benchmarks to assess the performance of different heuristic optimization methods. Let $C = \{1, 2, \ldots, n\}$ be a set of cities, and $D = (d_{i,j})_{n\times n}$ be a distance matrix that $d_{i,j}$ is the distance (or cost) between the ith and jth cities. Denote $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ as a permutation of $(1, 2, \ldots, n)$. A TSP can be formulated as

$$\min f(\pi) = \sum_{i=1}^{n-1} d_{\pi_i,\pi_{i+1}} + d_{\pi_n,\pi_1}. \tag{1}$$

In many applications, however, a salesman may consider not only the distance but also some other costs, such as time, risk, etc., simultaneously. In such cases, the problems become *multiobjective traveling salesman problems (MOTSPs)*. An MOTSP

* Corresponding author.
E-mail addresses: amzhou@cs.ecnu.edu.cn (A. Zhou), gxzhang@cs.ecnu.edu.cn (G. Zhang).

0898-1221/$ – see front matter
http://dx.doi.org/10.1016/j.camwa.2013.05.031

can be mathematically described as

$$\begin{aligned} \min \quad & F(\pi) = (f_1(\pi), f_2(\pi), \ldots, f_m(\pi)) \\ \text{s.t.} \quad & f_k(\pi) = \sum_{i=1}^{n-1} d_{\pi_i,\pi_{i+1}}(k) + d_{\pi_n,\pi_1}(k) \\ & k = 1, \ldots, m \end{aligned} \tag{2}$$

where m is the number of objectives (criteria), $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is a solution, i.e., a permutation of the city indices $(1, 2, \ldots, n)$, $D(k) = (d_{i,j}(k))_{n\times n}$ is the kth distance (cost) matrix, $f_k(\pi)$ is the kth objective, and $F(\pi)$ denotes the objective vector. Although there are many TSP test instances, there is still a lack of MOTSP benchmark problems.

Since the objectives in (2) usually conflict with each other, for example a tour with the shortest length might also be the most expensive, there does not exist a single tour that can minimize all the objectives simultaneously. Like other *multiobjective optimization problems (MOPs)*, a set of tradeoff solutions, which is called *Pareto set (PS)* in the decision space (*Pareto front (PF)* in the objective space) [2], defines the optimality of an MOTSP.

As in the case of TSP optimization, deterministic methods are not suitable for MOTSP optimization when the number of cities is huge. Heuristic methods are thus used to approximate the PS (PF) of an MOTSP. Among them, *evolutionary algorithms (EAs)* [3,4] are promising because they are able to approximate the whole PS (PF) in a single run. Actually, most of the *multiobjective evolutionary algorithms (MOEAs)* target to find a PS (PF) approximation which is as close to the true PS (PF) as possible and as diverse as possible [5]. In an MOEA, the reproduction operator and algorithm framework are the two key issues [6]. Some work related to the two issues in MOTSP optimization is summarized as follows.

- *Reproduction Operators for MOTSP:* the reproduction operators used for MOTSP optimization include: (1) neighborhood search: a new solution will be generated from the neighborhood of the parent [7,8]; (2) crossover and/or mutation operators: new solutions are generated by exchanging/mutating the components of the parents [9,10]; (3) probabilistic model based approaches: new solutions are sampled from probability models which are built to capture the population distribution. The quantum algorithm [11] and ant colony optimization (ACO) [12–15] based methods are using this idea; (4) local search strategies: in some work, the solutions are improved by local search methods [10,11,16–18].
- *MOEA Frameworks:* there are three MOEA frameworks which are widely used currently for dealing with generic MOPs: (1) MOEAs based on Pareto domination, which use the definition of Pareto optimality and density estimation strategies to select solutions [19]; (2) MOEAs based on performance indicators (IBEA), which apply the performance metrics to filter solutions [20]; and (3) MOEAs based on decomposition techniques (MOEA/D), which decompose a problem into a set of scalar objective sub-problems and tackle them at the same time [21,22].

Like ACO, *Estimation of distribution algorithms (EDAs)* use probabilistic models to extract the population distribution information and to sample new trial solutions [23]. Unlike the case of TSP test instances, a single probabilistic model may not work well for MOTSP instances. Thus how to maintain the probabilistic models plays a key role in most of current probabilistic model based methods for MOTSP. Since MOEA/D framework maintains a set of sub-problems, it is suitable for dealing with MOTSP test instances by using multiple probabilistic models [24]. In this paper, we propose to combine MOEA/D and EDA for dealing with MOTSP instances. In the new approach, named *multiobjective estimation of distribution algorithm based on decomposition (MEDA/D)*, an MOTSP is decomposed into a set of sub-problems (TSPs), and each sub-problem is with a probabilistic model. A probabilistic model contains both priori heuristic information and learned information from the population in the running process, and a new tour is directly sampled from the probabilistic model. By the cooperation of neighbor sub-problems, MEDA/D could approximate the PF of an MOTSP in a single run.

It should be noted that this paper is an extension of our previous work reported in [25]. The major contributions of the paper are as follows.

- A metric is defined to measure the degree of conflict (similarity) between two objectives in an MOTSP, and a new MOTSP test suite is therefore designed based on the conflict metric and some current TSP test instances.
- A multiobjective estimation of distribution algorithm based on decomposition is proposed to deal with MOTSP instances. Both priori heuristic information and learned information are utilized in a probabilistic model for each sub-problem of the algorithm.

The rest of the paper is organized as follows. Section 2 gives a definition of the conflict measurement between optimization objectives, and presents a new MOTSP test suite. Section 3 introduces the proposed algorithm MEDA/D in detail. The sub-problem definition and the probabilistic model building are addressed. Section 4 presents the comparison algorithms and performance metrics. In Section 5, MEDA/D is compared with two ACO based methods on three general MOTSP instances, and the sensitivity of MEDA/D to control parameters is also empirically studied. Section 6 shows the comparison results of three algorithms on the newly proposed MOTSP test suite. Finally, the paper is concluded in Section 7.

2. A new test suite for multiobjective TSP

Benchmark problems play a key role in the design of evolutionary algorithms. TSPLIB [26] is a well known library with different kind of problems. However, TSPLIB is only for single objective optimization. Therefore, some problems with the same number of cities are combined to formulate an MOTP. However, the properties of such kind of MOTSPs are hard to analyze since the relationship between these single objective TSP test instances is unknown. The purpose of this section is to present a general method to define MOTSP test problems, of which the similarity between different objectives is measurable in a sense, based on current single objective TSP test instances. To achieve this goal, we firstly give a definition of conflict measurement between two objectives, and then present the procedure to construct new MOTSP problems.

Definition 1 (*Conflict Measurement, Γ*)**.** The degree of conflict, denoted as $\Gamma(f_i, f_j)$, measures the similarity between objectives f_i and f_j in an MOP. It satisfies the following properties:

- $0 \le \Gamma(f_i, f_j) \le 1$;
- $\Gamma(f_i, f_j) = 0$ if there is no conflicts between f_i and f_j;
- $\Gamma(f_i, f_j) = 1$ if there is no similarity between f_i and f_j.

In the case of MOTSP, let $\pi^i = (\pi^i_1, \pi^i_2, \dots, \pi^i_n)$ be the optimal solution of objective f_i, and $\pi^j = (\pi^j_1, \pi^j_2, \dots, \pi^j_n)$ be the optimal solution of objective f_j. For simplicity, we also assume $\pi^i_1 = \pi^j_1$. We give the following definition of the conflict measurement for MOTSP.

$$\Gamma(f_i, f_j) = \frac{1}{n-1}\sum_{k=2}^{n} h(\pi^i_k - \pi^j_k)$$

where $h(x) = \begin{cases} 1 & x \neq 0 \\ 0 & x = 0. \end{cases}$

It is clear in the above definition, we use the difference between the optimal of tours of the objectives to represent the similarity between the objectives of an MOTSP. In the following, we use this definition to design new test instances.

Let $\pi = (\pi_1, \pi_2, \dots, \pi_n)$ be the optimal tour of a TSP, and $D = (d_{i,j})_{n\times n}$ be the distance matrix. Suppose that we want to construct an m-objective MOTSP, of which the conflict between every two objectives is γ, based on the given TSP, the procedure is as follows.

Step 1: Randomly choose m connected segments from π, denoted as $\pi(1), \pi(2), \dots, \pi(m)$, and the length of each segment is $\lfloor \frac{1}{2}(n-1)\gamma \rfloor$.

Step 2: Randomly permutate each segment, and get $\pi^*(1), \pi^*(2), \dots, \pi^*(m)$.

Step 3: For each objective k, $(k = 1, 2, \dots, m)$, use $\pi^*(1), \dots, \pi^*(i-1), \pi^*(i+1), \dots, \pi^*(m)$ to replace the corresponding parts in π, and get a new tour $\pi^k = (\pi^k_1, \pi^k_2, \dots, \pi^k_n)$, which shall be the optimal solution of the k^{th} objective.

Step 4: For each objective k, $(k = 1, 2, \dots, m)$, define the corresponding distance matrix $D(k) = (d_{i,j}(k))_{n\times n}$ as follows,

$$d_{\pi^k_i, \pi^k_j}(k) = d_{\pi_i, \pi_j}$$

where $i, j = 1, 2, \dots, n$.

Fig. 1 illustrates the basic idea to build an optimal tour for each objective from a given optimal tour in the cases of bi-objective and tri-objective MOTSP instances. From the figure, we can see that for each pair of objectives, there are only two different segments in the optimal tours. Therefore, the degree of conflict between each pair of objectives, (f_i, f_j), is

$$\Gamma(f_i, f_j) = \frac{1}{(n-1)}\left(\left\lfloor \frac{1}{2}(n-1)\gamma \right\rfloor + \left\lfloor \frac{1}{2}(n-1)\gamma \right\rfloor\right) = \gamma.$$

Fig. 2 shows an example of the constructed optimal tours in the case of a bi-objective MOTSP. The sub-tours are $\pi(1) = (3, 4)$ and $\pi(2) = (5, 6)$. The degree of conflict between the two objectives is $\Gamma(f_1, f_2) = \frac{2+2}{6-1} = 0.8$. Let the original distance matrix be

$$D = \begin{pmatrix} d_{1,1} & d_{1,2} & d_{1,3} & d_{1,4} & d_{1,5} & d_{1,6} \\ d_{2,1} & d_{2,2} & d_{2,3} & d_{2,4} & d_{2,5} & d_{2,6} \\ d_{3,1} & d_{3,2} & d_{3,3} & d_{3,4} & d_{3,5} & d_{3,6} \\ d_{4,1} & d_{4,2} & d_{4,3} & d_{4,4} & d_{4,5} & d_{4,6} \\ d_{5,1} & d_{5,2} & d_{5,3} & d_{5,4} & d_{5,5} & d_{5,6} \\ d_{6,1} & d_{6,2} & d_{6,3} & d_{6,4} & d_{6,5} & d_{6,6} \end{pmatrix},$$

the two new distance matrices for f_1 and f_2 will be

$$D(1) = \begin{pmatrix} d_{1,1} & d_{1,2} & \mathbf{d_{1,4}} & \mathbf{d_{1,3}} & d_{1,5} & d_{1,6} \\ d_{2,1} & d_{2,2} & \mathbf{d_{2,4}} & \mathbf{d_{2,3}} & d_{2,5} & d_{2,6} \\ \mathbf{d_{4,1}} & \mathbf{d_{4,2}} & \mathbf{d_{4,4}} & \mathbf{d_{4,3}} & \mathbf{d_{4,5}} & \mathbf{d_{4,6}} \\ \mathbf{d_{3,1}} & \mathbf{d_{3,2}} & \mathbf{d_{3,4}} & \mathbf{d_{3,3}} & \mathbf{d_{3,5}} & \mathbf{d_{3,6}} \\ d_{5,1} & d_{5,2} & \mathbf{d_{5,4}} & \mathbf{d_{5,3}} & d_{5,5} & d_{5,6} \\ d_{6,1} & d_{6,2} & \mathbf{d_{6,4}} & \mathbf{d_{6,3}} & d_{6,5} & d_{6,6} \end{pmatrix},$$

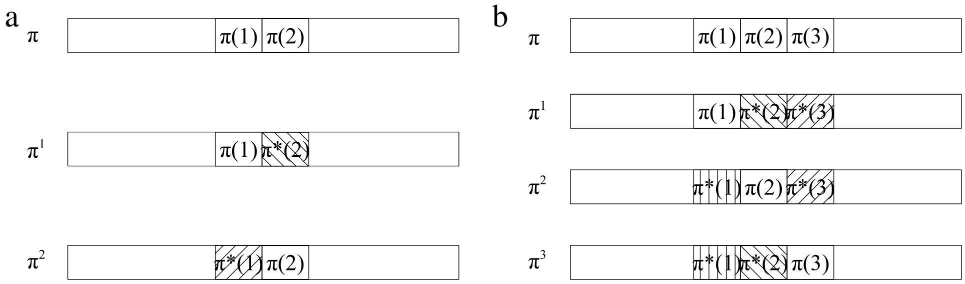


Fig. 1. An illustration of building an optimal tour for each objective from a given optimal tour: (a) bi-objective MOTSP, (b) tri-objective MOTSP.

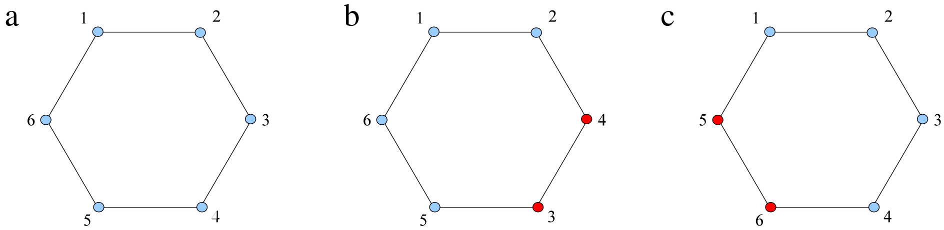


Fig. 2. An example of the constructed optimal tour in the case of a bi-objective MOTSP: (a) the original optimal tour $\pi = (1, 2, 3, 4, 5, 6)$, (b) the optimal tour of the 1th objective $\pi^1 = (1, 2, 4, 3, 5, 6)$, (c) the optimal tour of the 2nd objective $\pi^2 = (1, 2, 3, 4, 6, 5)$.

and

$$D(2) = \begin{pmatrix} d_{1,1} & d_{1,2} & d_{1,3} & d_{1,4} & \mathbf{d}_{1,\mathbf{6}} & \mathbf{d}_{1,\mathbf{5}} \\ d_{2,1} & d_{2,2} & d_{2,3} & d_{2,4} & \mathbf{d}_{\mathbf{2},\mathbf{6}} & \mathbf{d}_{\mathbf{2},\mathbf{5}} \\ d_{3,1} & d_{3,2} & d_{3,3} & d_{3,4} & \mathbf{d}_{\mathbf{3},\mathbf{6}} & \mathbf{d}_{\mathbf{3},\mathbf{5}} \\ d_{4,1} & d_{4,2} & d_{4,3} & d_{4,4} & \mathbf{d}_{\mathbf{4},\mathbf{6}} & \mathbf{d}_{\mathbf{4},\mathbf{5}} \\ \mathbf{d}_{\mathbf{6},\mathbf{1}} & \mathbf{d}_{\mathbf{6},\mathbf{2}} & \mathbf{d}_{\mathbf{6},\mathbf{3}} & \mathbf{d}_{\mathbf{6},\mathbf{4}} & \mathbf{d}_{\mathbf{6},6} & \mathbf{d}_{\mathbf{6},\mathbf{5}} \\ \mathbf{d}_{\mathbf{5},\mathbf{1}} & \mathbf{d}_{\mathbf{5},\mathbf{2}} & \mathbf{d}_{\mathbf{5},\mathbf{3}} & \mathbf{d}_{\mathbf{5},\mathbf{4}} & \mathbf{d}_{\mathbf{5},\mathbf{6}} & \mathbf{d}_{\mathbf{5},5} \end{pmatrix}.$$

It should be noted that to make each pair of objectives have the same γ value, it must satisfy

$$m \times \left\lfloor \frac{1}{2}(n-1)\gamma \right\rfloor \le n-1$$

i.e.,

$$m \times \gamma \le 2.$$

In the extreme case, if we want $\gamma = 1.0$ for each pair of objectives, we can randomly permutate the optimal tour of the original TSP to obtain an MOTSP.

Based on the above analysis, we produce 39 MOTSP test instances with 2, 3, 4 or 5 objectives from the TSPLIB. The details of the problems are shown in Table 1.

3. MEDA/D for multiobjective TSP

3.1. Basic idea

MOEA/D [21,22] is a general EA framework for dealing with MOPs. Like generic MOEAs, an MOEA/D starts with an initial population of candidate solutions; in an iteration, it generates some new trial solutions and selects the fittest ones to the next iteration; and it repeats the process until some termination conditions are satisfied. Unlike generic MOEAs, an MOEA/D decomposes a problem into a set of scalar objective sub-problems, and optimizes these sub-problems simultaneously. This idea is realized by solution cooperation in neighborhood, i.e., the solutions in the same neighborhood are used to generate new trial solutions, and the new trial solutions only update the old solutions in the same neighborhood. The following two notations are extremely important in an MOEA/D.

- *Sub-problem:* a multiobjective optimization problem is decomposed into a set of scalar objective problems and each of them is called a *sub-problem*. Hopefully, the optimal solution of the ith sub-problem $g^i(\pi)$ lies in the PS (PF) of the original problem.

Table 1
A new MOTSP test suite based on TSPLIB.

Instance name	#Objectives (m)	#Cities (n)	Γ (γ)	Instance name	#Objectives (m)	#Cities (n)	Γ (γ)
kroA100-2-0.2	2	100	0.2	kroA100-3-0.2	3	100	0.2
kroA100-2-0.4	2	100	0.4	kroA100-3-0.4	3	100	0.4
kroA100-2-0.6	2	100	0.6	kroA100-3-0.6	3	100	0.6
kroA100-2-1.0	2	100	1.0	kroA100-3-1.0	3	100	1.0
ch150-2-0.2	2	150	0.2	ch150-3-0.2	3	150	0.2
ch150-2-0.4	2	150	0.4	ch150-3-0.4	3	150	0.4
ch150-2-0.6	2	150	0.6	ch150-3-0.6	3	150	0.6
ch150-2-1.0	2	150	1.0	ch150-3-1.0	3	150	1.0
gr202-2-0.2	2	202	0.2	gr202-3-0.2	3	202	0.2
gr202-2-0.4	2	202	0.4	gr202-3-0.4	3	202	0.4
gr202-2-0.6	2	202	0.6	gr202-3-0.6	3	202	0.6
gr202-2-1.0	2	202	1.0	gr202-3-1.0	3	202	1.0
kroA100-4-0.2	4	100	0.2	kroA100-5-0.2	5	100	0.2
kroA100-4-0.4	4	100	0.4	kroA100-5-1.0	5	100	0.4
kroA100-4-1.0	4	100	1.0				
ch150-4-0.2	4	150	0.2	ch150-5-0.2	5	150	0.2
ch150-4-0.4	4	150	0.4	ch150-5-1.0	5	150	1.0
ch150-4-1.0	4	150	1.0				
gr202-4-0.2	4	202	0.2	gr202-5-0.2	5	202	0.2
gr202-4-0.4	4	202	0.4	gr202-5-1.0	5	202	1.0
gr202-4-1.0	4	202	1.0				

- *Neighborhood:* the neighborhood $B^i = (i_1, i_2, \ldots, i_K)$ of the ith sub-problem contains the indices of similar sub-problems, i.e., the i_jth ($j = 1, \ldots, K$) sub-problems are the most similar ones to the ith sub-problem.

Fig. 3 illustrates the basic idea of an MOEA/D. A bi-objective MOP is decomposed into 10 sub-problems $\{g^1, \ldots, g^{10}\}$ and the neighborhood of the 5th sub-problem contains the 3rd, 4th, 5th, 6th, and 7th sub-problems.

3.2. Sub-problem definition

In this paper, we use the following Tchebycheff approach to define the sub-problems.

$$\min g^i(\pi) = g(\pi|\lambda^i, z^*) = \max_{1\le j\le m} \lambda^i_j |f_j(\pi) - z^*_j| \tag{3}$$

where $\lambda^i = (\lambda^i_1, \ldots, \lambda^i_m)^T$ is a weight vector with the ith sub-problem, $z^* = (z^*_1, \ldots, z^*_m)^T$ is a reference point. It is clear that all the sub-problems are with the same form and can only be differentiated by the weight vectors. If two vectors are close to each other, the corresponding sub-problems should be similar to each other and their optima should also be close in both the decision and objective spaces in most cases. By using the weight vectors, the neighborhood could be determined before the algorithm execution. It should be noticed that it is not necessary that all sub-problems have the same form.

In MEDA/D, the ith ($i = 1, \ldots, N$) sub-problem is with

- a scalar objective problem $\min g^i(\pi)$ defined in (3),
- a decision vector π^i and its objective vector $F^i = F(\pi^i)$,
- a set of indices B^i of the neighborhood sub-problems which are with the K closest weight vectors to λ^i, and
- a probability model P^i which stores the information of the ith sub-problem.

3.3. Probabilistic model building

A probabilistic model P^i stores the information extracted from the population for the ith sub-problem. Like ACO based approaches, P^i contains both the prior heuristic and learned distribution information.

Without other information, it is natural to select the next city from some closest cities. Thus the distances between cities could be used as priori information. However, in the case of MOTSP, the distance matrices are only available for the extreme sub-problems, i.e., the sub-problems with the objective functions as their sub-problem objectives. For the ith sub-problem, we define a pseudo distance matrix as

$$D^i = \sum_{k=1}^{m} \lambda^i_k D(k). \tag{4}$$

It is clear that the pseudo distance between the sth and tth cities is $d^i_{s,t} = \sum_{k=1}^{m} \lambda^i_k d_{s,t}(k)$ which is sub-problem dependent.

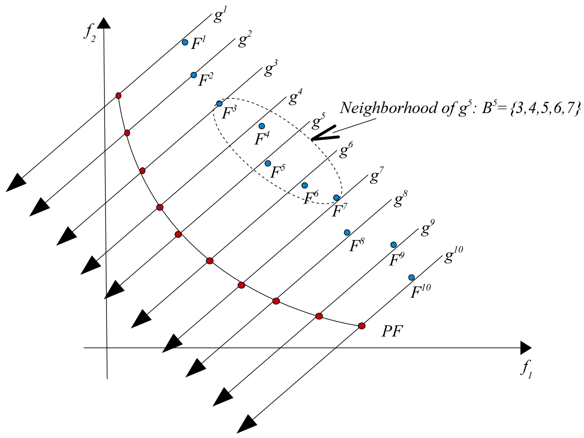


Fig. 3. An illustration of the basic idea of MOEA/D.

Let $Q^i = (q^i_{s,t})_{n\times n}$ be a matrix which represents the learned information for the ith sub-problem, where $q^i_{s,t}$ denotes the connection strength between the sth and tth cities. Let

$$q_0 = \min_{s,t=1,\dots,n,k=1,\dots,m} d_{s,t}(k),$$

i.e., q_0 is the minimum distance between any pair of cities in all the given distance matrices. The learned information Q^i is initialized as

$$q^i_{s,t} = q_0 \tag{5}$$

for all $s, t = 1, \dots, n$ and $i = 1, \dots, N$. In the running process, once the ith sub-problem is updated by a new trial solution π, its learned information matrix Q^i is updated as

$$q^i_{s,t} = \begin{cases} (1-\rho)q^i_{s,t} + \rho q_0 & \text{if } (s,t) \text{ is in } \pi \\ (1-\rho)q^i_{s,t} & \text{otherwise} \end{cases} \tag{6}$$

where ρ denotes the learning rate.

Based on the above pseudo distance matrix D^i and learned information matrix Q^i, we define the probability matrix P^i for the ith sub-problem as

$$P^i = (p^i_{s,t})_{n\times n} = \left(\frac{\left(q^i_{s,t}/d^i_{s,k}\right)^\alpha}{\sum_{k=1}^{n} \left(q^i_{s,k}/d^i_{s,k}\right)^\alpha} \right)_{n\times n} \tag{7}$$

where α balances the contributions of the priori and learned information; $p^i_{s,t}$ denotes the probability that the sth and tth cities are connected in the route of the ith sub-problem.

3.4. Algorithm framework

The main framework of MEDA/D is as follows.

Step 1 Initialization:

1.1 Convert an MOTSP into N sub-problems by (3) and randomly generate a solution for each sub-problem. Initialize the reference point z^* as

$$z^*_k = \arg\min_{i=1,\dots,N} f_k(\pi^i)$$

for $k = 1, \dots, m$. Initialize the weight vectors λ^i which are well distributed.

1.2 Initialize the neighborhood B^i for each sub-problem i.

1.3 Initialize the probability matrix P^i for each sub-problem i as (7) in which $d^i_{s,t}$ is estimated as in (4) and $q^i_{s,t}$ is initialized as in (5).

Step 2 Sample and Update: For each sub-problem $i = 1, \ldots, N$, do

2.1 Sample a new solution $\pi = (\pi_1, \ldots, \pi_n)$: Let $C = \{1, \ldots, n\}$ be the unvisited cities. The first city π_1 is set to be $\pi_1 = 1$ and let $C = C/\{\pi_1\}$. Then the next city π_j $(j > 1)$ is randomly selected according to the probability P^i, i.e.,

$$\pi_j \sim \frac{p^i_{\pi_{j-1},\pi_j}}{\sum_{\pi_k \in C} p^i_{\pi_{j-1},\pi_k}}$$

where $\pi_j \in C$. Once π_j is chosen, set $C = C/\{\pi_j\}$. Repeat the process until the whole tour is constructed.

2.2 Update of reference point: For each $k = 1, \ldots, m$, if $z^*_k > f_k(\pi)$, then set $z^*_k = f_k(\pi)$.

2.3 Update solutions: Set $A^i = \{k | k \in B^i, g^k(\pi) < g^k(\pi^k)\}$, i.e., A^i denotes all the neighbor solutions which are worse than the newly sampled solution π according to the corresponding sub-problem definitions. Randomly update at most μ solutions in A^i by π.

2.4 Update probability model: For each sub-problem which is updated in the above step, update its corresponding learned information matrix Q as in (6).

Step 3 Stopping Criterion: If the stop conditions are satisfied, then stop; otherwise go to **Step 2**.

We would like to make some comments to the algorithm.

- In **Step 1.1**, the weight vectors are generated to make them well distributed. In the case of bi-objective problems, the weight $\lambda^i = \left(\frac{i-1}{N-1}, \frac{N-i}{N-1}\right)$ for $i = 1, \ldots, N$. The method to generate weighs for problems with more than 2 objectives is referred to [21] for more details.
- In **Step 3**, the algorithm stops after a given maximum number of generations.

4. Comparison algorithms and performance metrics

4.1. Comparison algorithms

In this paper, we compare MEDA/D with two ACO based algorithms, MOACO [13] and MOEA/D-ACO [24].

MOACO is a modified version of an ant colony optimization (ACO) algorithms for bi-objective TSPs [13]. In the original algorithm, each ant makes its decision according to the following probabilities (Eq. (3) in [13]),

$$p_{ij} = \frac{\tau_{ij}^{\lambda\alpha} \cdot \tau_{ij}^{\prime(1-\lambda)\alpha} \cdot \eta_{ij}^{\lambda\alpha} \cdot \eta_{ij}^{\prime(1-\lambda)\alpha}}{\Sigma_{h\in S} \tau_{ih}^{\lambda\alpha} \cdot \tau_{ih}^{\prime(1-\lambda)\alpha} \cdot \eta_{ih}^{\lambda\alpha} \cdot \eta_{ih}^{\prime(1-\lambda)\alpha}}.$$

To make it work for arbitrary objective problems, we slightly change the probabilities to

$$p_{ij} = \frac{\Pi_{k=1}^{m}(\tau_{k_{ij}}^{\lambda_k\alpha} \cdot \eta_{k_{ij}}^{\lambda_k\alpha})}{\Sigma_{h\in S} \Pi_{k=1}^{m}(\tau_{k_{ih}}^{\lambda_k\alpha} \cdot \eta_{k_{ih}}^{\lambda_k\alpha})}$$

where λ_k is the weight with the kth objective. Each ant is with a weight vector and all the weight vectors are defined as the same in MEDA/D. More details of MOACO are referred to [13].

MOEA/D-ACO is a state-of-the-art algorithm for dealing with discrete multiobjective optimization problems [24]. It utilizes the ACO algorithm in the framework of MOEA/D and each ant will search for a sub-problem in MOEA/D. Unlike MEDA/D, all the ants are divided into several groups and a probability model is maintained for each ant group. More details of MOEA/D-ACO are referred to [24].

4.2. Performance metrics

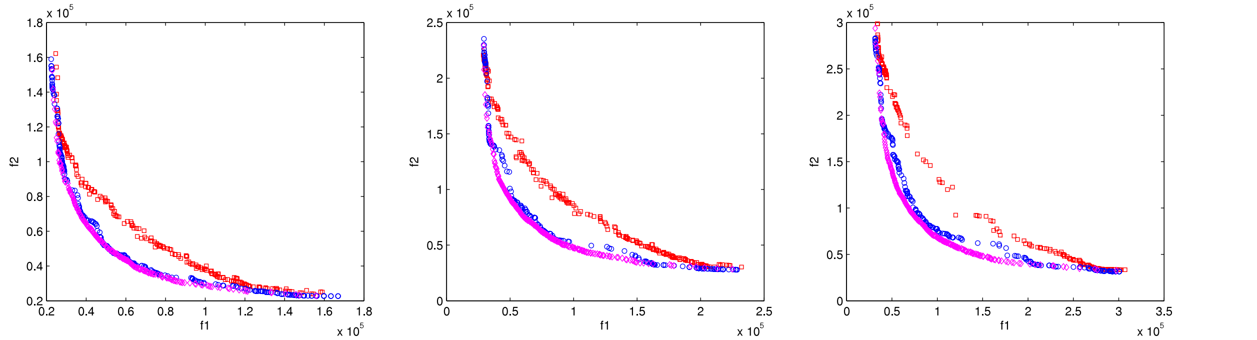
To have a fair comparison, the Coverage (C) [27] and Inverted Generational Distance (IGD) [28] metrics are used to access the performance of the two algorithms. The Coverage metric is used to compare the achieved non-dominated solutions.

$$C(P_1, P_2) = \frac{|\{\pi | \pi \in P_2, \exists \phi \in P_1 : F(\phi) \prec F(\pi)\}|}{|P_2|}, \tag{8}$$

where P_1 and P_2 are the obtained non-dominated sets by two algorithms, $F(\phi) \prec F(\pi)$ denotes $F(\phi)$ dominates $F(\pi)$. $C(P_1, P_2)$ is not necessarily equal to $1 - C(P_2, P_1)$. If $C(P_1, P_2)$ is large and $C(P_2, P_1)$ is small, then P_1 is better than P_2 in a sense.

Table 2
Statistical results (mean ± std.) of MOACO(A), MOEA/D-ACO(D), and MEDA/D(E) over 50 runs.

Instance	Coverage				*IGD*		
	C(A, E)	C(E, A)	C(D, E)	C(E, D)	A	D	E
kroAB100	0.000 ± 0.001	**0.997** ± 0.007	0.099 ± 0.051	**0.510** ± 0.102	11 711.918 ± 476.527	2114.272 ± 287.233	**1171.542** ± 116.073
kroAB150	0.001 ± 0.002	**0.974** ± 0.029	0.030 ± 0.014	**0.664** ± 0.083	21 472.019 ± 581.964	4934.916 ± 571.815	**2119.730** ± 333.939
kroAB200	0.002 ± 0.003	**0.958** ± 0.043	0.027 ± 0.013	**0.697** ± 0.094	30 354.882 ± 618.923	8209.367 ± 613.607	**2628.479** ± 383.955



(a) KroAB100. (b) KroAB150. (c) KroAB200.

Fig. 4. The best approximations obtained by the three algorithms according to the *IGD* values over 50 runs. The red squares are with MOACO, the blue circles are with MOEA/D-ACO, and the magenta diamonds are with MEDA/D.

Let P^* be a set of uniformly distributed Pareto optimal points in the PF. Let P be an approximation to the PF. The *IGD* metric is defined as follows,

$$IGD(P^*, P) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|}$$

where $d(v, P)$ is a minimum distance between v and any point in P, and $|P^*|$ is the cardinality of P^*. The *IGD* metric can measure both convergence and diversity. To have a low *IGD* value, P must be close to the PF and cannot miss any part of the whole PF. In our experiments, all the true PFs are unknown. Therefore, we combine all the obtained results in all runs by all algorithms and find out the non-dominated solutions from the combination as the reference P^*. If the size of P^* is more than 5000, we randomly select 5000 non-dominated solutions as P^*.

5. Experimental results on general MOTSP

We select three test problems from the TSPLIB [26] and combine the distance matrices of two TSP test instances with the same number of cities to construct three bi-objective problems: KroAB100, KroAB150, and KroAB200. The name of the problems is from the two corresponding TSP test instances. For example, KroAB100 represents the combination of KroA100 (the 1st objective) and KroB100 (the 2nd objective).

The three algorithms, MOACO, MOEA/D-ACO and MEDA/D, are all implemented by C++ and executed in the same environment. For the three algorithms, the population size is set to be 250, and the number of generations is 1000 for all problems. The parameters of MEDA/D are as follows: $\alpha = 3$, $\rho = 0.1$, $K = 20$ and $\mu = 2$. The other parameters of MOACO are the same as in [13], and the other parameters of MOEA/D-ACO are the same as in [24]. The statistical results are based on 50 independent runs.

5.1. Statistical results

The best approximations obtained by MOACO, MOEA/D-ACO and MEDA/D are plotted in Fig. 4. It is clear that MEDA/D outperforms MOACO on all the given test instances, and MOEA/D-ACO and MEDA/D have similar results. The solutions obtained by MEDA/D are more close to the PF and have better distributions than those obtained by MOEA/D-ACO especially in the middle areas. In the extreme areas, both MOEA/D-ACO and MEDA/D could get similar results but the distributions of the solutions obtained by MEDA/D are better than those obtained by MOEA/D-ACO.

Table 2 presents the statistical results of the Coverage and *IGD* metrics. It shows that on average, at least 95.8% solutions obtained by MOACO and 51.0% solutions obtained by MOEA/D-ACO are dominated by those obtained by MEDA/D. On the contrary, however only at most 0.2% and 9.9% solutions obtained by MEDA/D are dominated by those obtained by MOACO and MOEA/D-ACO respectively. This indicates that according to the coverage metric, MEDA/D performs much better than MOACO. The *IGD* values represent both the diversity and convergence qualities of the final approximations. It can be seen

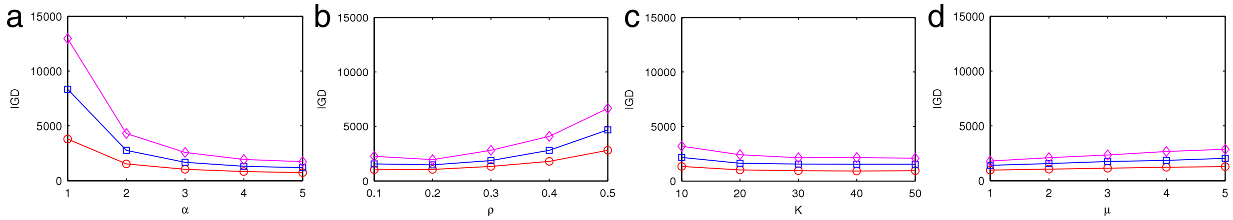


Fig. 5. Sensitivity to MEDA/D parameters on KroAB100 (circles), KroAB150 (squares), and KroAB200 (diamonds).

from Table 2 that for all the problems, the approximations obtained by MEDA/D are better than those of MOACO and MOEA/D-EDA.

It is clear that both MOEA/D-ACO and MEDA/D outperform MOACO, and the reason might be that the MOEA/D framework is more suitable for MOTSPs comparing to traditional MOEA frameworks. One of the major differences between MOEA/D-ACO and MEDA/D is that a probability model is built for each sub-problem in MEDA/D while a probability model is built for a group of sub-problems in MOEA/D-ACO. Therefore, the model quality in MEDA/D might be higher than MOEA/D-ACO, and this might be the reason why MEDA/D is slightly better than MOEA/D-ACO. It is assumed in MOEA/D framework that a well distributed weight vectors will lead to a well distributed solutions. However, this assumption does not hold in many cases because the final solution distribution depends not only on the weights but also on the geometry shape of the PF. We can see from Fig. 4 that the solutions obtained by MOEA/D-ACO and MEDA/D distribute more dense around the middle area of the PF. Adaptively turing the weight online might be able to solve this problem [8,29], and it is worth trying in the future.

5.2. Sensitivity to algorithm parameters

In MEDA/D, there are four parameters: α which balances the priori and learned information in the probabilistic model; learning rate ρ; neighborhood size K; and maximum number updated solutions μ. In this section, we study the sensitivities of MEDA/D on these parameters. The parameters are set as $\alpha = 1, 2, 3, 4, 5$, $\rho = 0.1, 0.2, 0.3, 0.4, 0.5$, $K = 10, 20, 30, 40, 50$, and $\mu = 1, 2, 3, 4, 5$. In the experiments, we only change one parameter and the other parameters are the same as in the previous section.

The average IGD values over the change of parameters are plotted in Fig. 5. We can see that (1) the performance of MEDA/D increases slightly as α increases; (2) the performance of MEDA/D decreases slightly as ρ increases; (3) K and μ, do not influence MEDA/D much. Overall, although α and ρ influence the performances of MEDA/D, our new approach is not sensitivity to the algorithm parameters.

6. Experimental results on the proposed MOTSP test suite

In this section, we compare MEDA/D with MOACO and MOEA/D-ACO on the newly proposed MOTSP test suite. The population size is 250 for 2-objective problems, 276 for 3-objective problems, 286 for 4-objective problems, and 330 for 5-objective problems. The algorithm parameters are the same as in Section 5.

Table 3 shows the statistical results of Coverage and *IGD* metric values that obtained by the three algorithms over 50 runs. For the Coverage metric, MEDA/D outperforms MOACO on all the 39 problems, and MEDA/D works better than MOEA/D-ACO on 33 problems. According to the *IGD* metric values, MOEA/D-ACO obtains best results on 10 problems and MEDA/D works the best on the other 29 problems. It is clear that MEDA/D works better than MOACO and MOEA/D-ACO on most of the problems and the reason might be the same as in the previous section.

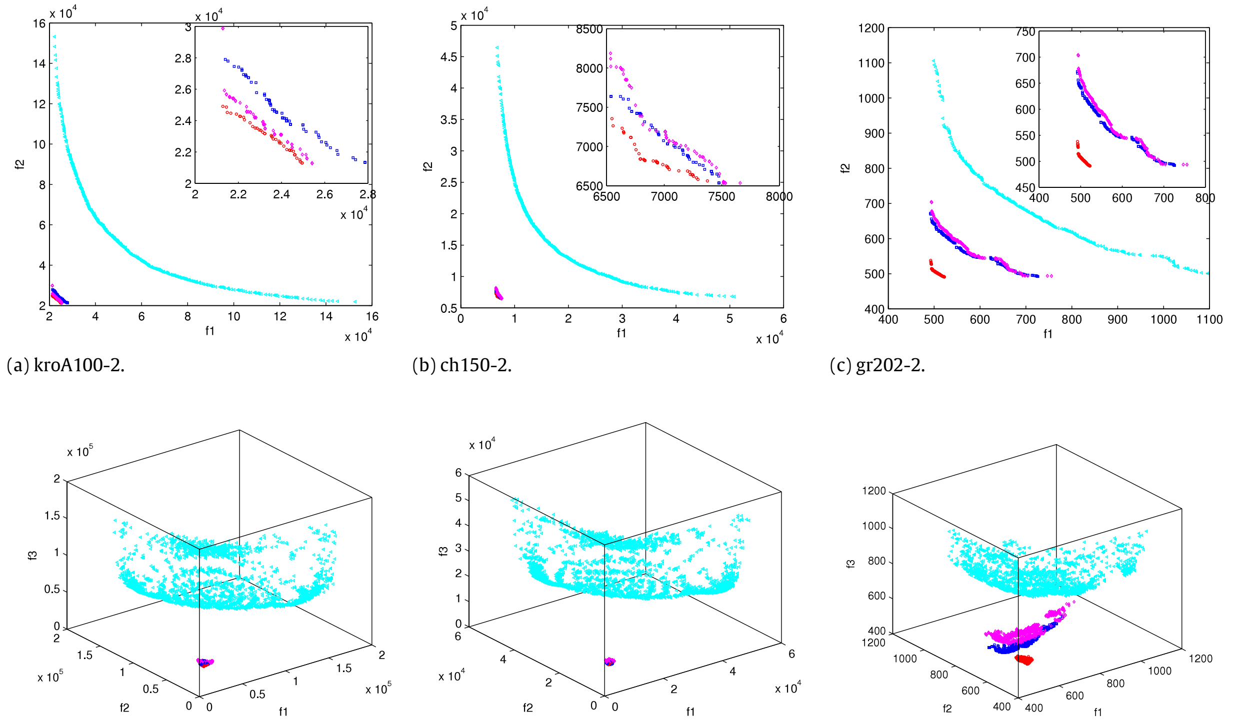
Fig. 6 plots the non-dominated fronts of all the 50 runs obtained by MEDA/D on the six problems. We can see that as γ increases, the covered range of the PF increases as well. Therefore, we can conclude that the parameter γ controls the conflict (similarity) between the objectives in a sense. It is clear that the objective conflict degree will influence the algorithm performance. In this paper, MEDA/D does not take this issue into consideration. However, the conflict measurement may help us to analyze the properties of an MOTSP, to design problem-specific algorithms, and it is worth for future investigating.

7. Conclusions

In this paper, we firstly gave the definition for the conflict measurement between optimization objectives. Based on this definition, we designed a new test suite for multiobjective TSP. Secondly, we proposed a *multiobjective estimation of distribution algorithm based on decomposition (MEDA/D)* for the MOTSP instances. In the framework of MEDA/D, an MOTSP was decomposed into a set of scalar objective sub-problems. A probabilistic model, which uses both priori heuristic and learned information, was built to store information and to sample new trial solutions for each sub-problem. The new approach was compared with MOACO [13] and MOEA/D-ACO [24] on three general MOTSP instances and the newly designed MOTSP test suite. The statistical results show that (1) the new benchmark problems are suitable for assessing

Table 3
Statistical results (mean ± std.) of MOACO(A), MOEA/D-ACO(D), and MEDA/D(E) over 50 runs. +(−) supports (fails to support) the hypothesis that MOACO and MOEA/D-ACO performs better than MEDA/D at the 95% significance level under *t*-test.

Instance	Coverage				*IGD*		
	C(A, E)	C(E, A)	C(D, E)	C(E, D)	A	D	E
kroA100-2-0.2	0.000 ± 0.000	**1.000** ± 0.000	**0.845** ± 0.237	0.091 ± 0.197	2864.012 ± 507.822 (−)	**206.483** ± 141.112 (+)	414.516 ± 149.998
kroA100-2-0.4	0.000 ± 0.000	**1.000** ± 0.000	**0.676** ± 0.236	0.161 ± 0.170	2218.085 ± 291.144 (−)	**203.792** ± 147.128 (+)	315.931 ± 89.977
kroA100-2-0.6	0.000 ± 0.000	**1.000** ± 0.000	**0.767** ± 0.330	0.195 ± 0.319	2414.171 ± 69.265 (−)	**148.418** ± 172.527 (+)	239.057 ± 103.704
kroA100-2-1.0	0.000 ± 0.001	**0.997** ± 0.010	0.105 ± 0.043	**0.433** ± 0.098	10 491.392 ± 625.161 (−)	2340.290 ± 320.107 (−)	**1428.807** ± 189.034
ch150-2-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.255 ± 0.271	**0.470** ± 0.283	299.492 ± 79.791 (−)	79.796 ± 30.892 (−)	**58.755** ± 16.808
ch150-2-0.4	0.001 ± 0.005	**0.994** ± 0.020	0.352 ± 0.230	**0.388** ± 0.231	321.145 ± 58.391 (−)	80.675 ± 32.069 (−)	**65.501** ± 24.993
ch150-2-0.6	0.002 ± 0.010	**0.987** ± 0.053	0.347 ± 0.259	**0.439** ± 0.284	267.591 ± 48.107 (−)	67.415 ± 19.263 (−)	**53.246** ± 13.615
ch150-2-1.0	0.000 ± 0.001	**0.988** ± 0.021	0.050 ± 0.025	**0.568** ± 0.085	4031.377 ± 126.245 (−)	1108.062 ± 94.140 (−)	**423.670** ± 60.669
gr202-2-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.078 ± 0.166	**0.853** ± 0.251	61.939 ± 8.995 (−)	10.777 ± 5.146 (−)	**4.034** ± 1.610
gr202-2-0.4	0.000 ± 0.000	**1.000** ± 0.000	0.141 ± 0.146	**0.643** ± 0.253	51.790 ± 11.632 (−)	16.566 ± 4.326 (−)	**8.712** ± 2.368
gr202-2-0.6	0.000 ± 0.000	**1.000** ± 0.000	0.150 ± 0.118	**0.630** ± 0.216	64.484 ± 5.558 (−)	17.047 ± 5.407 (−)	**8.760** ± 2.347
gr202-2-1.0	0.000 ± 0.000	**1.000** ± 0.000	0.070 ± 0.044	**0.666** ± 0.132	70.112 ± 5.233 (−)	36.043 ± 4.393 (−)	**15.550** ± 2.813
kroA100-3-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.050 ± 0.077	**0.819** ± 0.151	3314.287 ± 743.751 (−)	768.225 ± 162.685 (−)	**385.092** ± 78.897
kroA100-3-0.4	0.000 ± 0.000	**1.000** ± 0.000	0.083 ± 0.100	**0.801** ± 0.178	3301.052 ± 340.244 (−)	734.466 ± 206.749 (−)	**372.343** ± 60.216
kroA100-3-0.6	0.000 ± 0.000	**0.981** ± 0.024	0.194 ± 0.084	**0.217** ± 0.081	1910.299 ± 1019.078 (−)	**205.918** ± 39.606 (+)	232.535 ± 20.038
kroA100-3-1.0	0.001 ± 0.001	**0.800** ± 0.035	0.125 ± 0.020	**0.167** ± 0.020	18 836.544 ± 383.444 (−)	**3434.648** ± 71.623 (+)	7632.776 ± 227.533
ch150-3-0.2	0.000 ± 0.000	**0.986** ± 0.037	0.026 ± 0.038	**0.774** ± 0.168	477.534 ± 90.271 (−)	256.218 ± 43.316 (−)	**121.975** ± 42.654
ch150-3-0.4	0.002 ± 0.007	**0.977** ± 0.036	0.037 ± 0.039	**0.745** ± 0.154	333.585 ± 46.626 (−)	196.654 ± 31.279 (−)	**86.697** ± 19.050
ch150-3-0.6	0.003 ± 0.006	**0.980** ± 0.032	0.057 ± 0.067	**0.640** ± 0.200	320.556 ± 50.134 (−)	218.755 ± 41.785 (−)	**102.267** ± 16.018
ch150-3-1.0	0.006 ± 0.004	**0.703** ± 0.029	0.128 ± 0.025	**0.167** ± 0.021	6318.189 ± 141.098 (−)	**1113.392** ± 19.739 (+)	2281.836 ± 49.574
gr202-3-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.001 ± 0.010	**0.984** ± 0.060	83.141 ± 16.272 (−)	27.130 ± 5.337 (−)	**8.159** ± 2.669
gr202-3-0.4	0.000 ± 0.000	**1.000** ± 0.000	0.005 ± 0.015	**0.937** ± 0.094	69.503 ± 8.045 (−)	33.122 ± 4.364 (−)	**10.984** ± 2.429
gr202-3-0.6	0.000 ± 0.000	**0.996** ± 0.014	0.009 ± 0.015	**0.833** ± 0.110	49.062 ± 4.321 (−)	28.533 ± 2.439 (−)	**13.107** ± 2.079
gr202-3-1.0	0.000 ± 0.001	**0.934** ± 0.044	0.024 ± 0.019	**0.495** ± 0.078	85.038 ± 3.761 (−)	41.100 ± 1.554 (−)	**29.300** ± 1.733
kroA100-4-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.000 ± 0.000	**1.000** ± 0.000	3254.677 ± 760.586 (−)	3938.343 ± 396.353 (−)	**380.979** ± 64.564
kroA100-4-0.4	0.000 ± 0.000	**1.000** ± 0.000	0.000 ± 0.000	**1.000** ± 0.000	3842.058 ± 379.294 (−)	3649.730 ± 232.077 (−)	**397.711** ± 42.138
kroA100-4-1.0	0.001 ± 0.002	**0.375** ± 0.045	0.060 ± 0.014	**0.069** ± 0.006	24 113.723 ± 212.579 (−)	**5896.570** ± 211.954 (+)	18 711.767 ± 142.378
ch150-4-0.2	0.000 ± 0.000	**0.995** ± 0.026	0.000 ± 0.000	**1.000** ± 0.000	509.265 ± 102.748 (−)	1099.647 ± 151.858 (−)	**99.576** ± 12.893
ch150-4-0.4	0.000 ± 0.001	**0.979** ± 0.046	0.000 ± 0.000	**1.000** ± 0.000	294.115 ± 66.714 (−)	1221.478 ± 186.309 (−)	**63.328** ± 14.743
ch150-4-1.0	0.004 ± 0.003	**0.184** ± 0.030	**0.138** ± 0.021	0.041 ± 0.004	8614.818 ± 89.365 (−)	**1696.637** ± 73.584 (+)	6134.056 ± 55.312
gr202-4-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.000 ± 0.000	**1.000** ± 0.000	78.414 ± 10.342 (−)	119.162 ± 10.107 (−)	**12.353** ± 1.946
gr202-4-0.4	0.000 ± 0.000	**0.999** ± 0.004	0.000 ± 0.000	**1.000** ± 0.000	62.522 ± 7.052 (−)	113.815 ± 11.123 (−)	**17.426** ± 2.704
gr202-4-1.0	0.001 ± 0.002	**0.404** ± 0.161	0.000 ± 0.000	**1.000** ± 0.001	73.601 ± 3.198 (−)	120.161 ± 4.196 (−)	**42.800** ± 1.511
kroA100-5-0.2	0.000 ± 0.000	**1.000** ± 0.000	0.313 ± 0.261	**0.340** ± 0.263	3472.432 ± 797.843 (−)	509.710 ± 134.440 (−)	**428.410** ± 81.736
kroA100-5-1.0	0.001 ± 0.002	**0.153** ± 0.027	**0.027** ± 0.009	0.024 ± 0.005	30 412.946 ± 148.010 (−)	**16 736.090** ± 194.153 (+)	21 680.005 ± 150.901
ch150-5-0.2	0.000 ± 0.000	**1.000** ± 0.001	0.147 ± 0.158	**0.379** ± 0.214	557.810 ± 92.964 (−)	162.270 ± 47.758 (−)	**127.354** ± 13.471
ch150-5-1.0	0.005 ± 0.003	**0.110** ± 0.028	**0.033** ± 0.008	0.020 ± 0.007	10 238.064 ± 78.568 (−)	**5710.382** ± 54.928 (+)	6444.951 ± 45.496
gr202-5-0.2	0.000 ± 0.000	**0.996** ± 0.013	0.005 ± 0.009	**0.641** ± 0.232	66.361 ± 12.813 (−)	31.561 ± 5.435 (−)	**16.979** ± 2.167
gr202-5-1.0	0.001 ± 0.002	**0.203** ± 0.087	0.009 ± 0.009	**0.538** ± 0.043	103.498 ± 3.578 (−)	86.176 ± 3.231 (−)	**70.444** ± 2.440



(a) kroA100-2. (b) ch150-2. (c) gr202-2.

(d) kroA100-3. (e) ch150-3. (f) gr202-3.

Fig. 6. The non-dominated front of all the 50 runs obtained by MEDA/D on the six groups of problems. The circles are with $\gamma = 0.2$, squares are with $\gamma = 0.4$, diamonds are with $\gamma = 0.6$, and triangles are with $\gamma = 1.0$.

the performance of an algorithm; (2) the proposed method MEDA/D outperforms MOACO and MOEA/D-ACO on the most of the given test instances; (3) MEDA/D is not sensible to the algorithm parameters.

Discrete optimization problems are usually harder than continuous optimization problems due to their complicated properties. This paper defined a metric to measure the degree of conflict between objectives, which may help us to do analysis. The work reported in this paper is preliminary, and there are several directions for future work: (1) improving the performance of current MOEAs by considering the objective conflict, and (2) extending the idea to construct other multiobjective discrete problems such as 0/1 Knapsack problems and routing problems.

Acknowledgments

The authors would like to thank Dr. L. Ke for providing the source code of MOEA/D-ACO. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2011CB707104 and the National Natural Science Foundation of China under Grant 61273313.

References

[1] R. Baraglia, J. Hidalgo, R. Perego, A hybrid heuristic for the traveling salesman problem, IEEE Transactions on Evolutionary Computation 5 (6) (2001) 613–622.
[2] K. Miettinen, Nonlinear Multiobjective Optimization, Springer, 1999.
[3] K.C. Tan, E.F. Khor, T.H. Lee, Multiobjective Evolutionary Algorithms and Applications, Springer, 2005.
[4] C.A. Coello Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problem, Kluwer, 2002.
[5] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons Ltd., 2001.
[6] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: a survey of the state of the art, Swarm and Evolutionary Computation 1 (1) (2011) 32–49.
[7] H. Li, D. Landa-Silva, Evolutionary multi-objective simulated annealing with adaptive and competitive search direction, in: IEEE Congress on Evolutionary Computation, CEC 2008, 2008, pp. 3311–3318.
[8] H. Li, D. Landa-Silva, An adaptive evolutionary multi-objective approach based on simulated annealing, Evolutionary Computation 19 (4) (2011) 561–595.
[9] Z. Yan, L. Zhang, L. Kang, G. Lin, A new MOEA for multi-objective TSP and its convergence property analysis, in: Evolutionary Multi-Criterion Optimization, EMO 2003, 2003, pp. 342–354.
[10] W. Peng, Q. Zhang, H. Li, Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem, in: Multi-Objective Memetic Algorithms, Springer, 2009, pp. 309–324.
[11] H. Li, D. Landa-Silva, X. Gandibleux, Evolutionary multi-objective optimization algorithms with probabilistic representation based on pheromone trails, in: IEEE Congress on Evolutionary Computation, CEC 2010, 2010, pp. 1–8.

[12] L.M. Gambardella, M. Dorigo, Ant-Q: a reinforcement learning approach to the traveling salesman problem, in: Twelfth International Conference on Machine Learning, ML-95, 1995, pp. 252–260.
[13] S. Iredi, D. Merkle, M. Middendorf, Bi-criterion optimization with multi colony ant algorithms, in: Evolutionary Multi-Criterion Optimization, EMO 2001, Springer, 2001, pp. 359–372.
[14] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, C. Stummer, Pareto ant colony optimization: a metaheuristic approach to multiobjective portfolio selection, Annals of Operations Research 131 (1) (2004) 79–99.
[15] P. Cardoso, M. Jesus, A. Marquez, MONACO—multi-objective network optimisation based on an ACO, in: Proceedings of X Encuentros de Geometria Computacional, 2003.
[16] A. Jaszkiewicz, On the performance of multiple-objective genetic local search on the 0/1 knapsack problem—a comparative experiment, IEEE Transactions on Evolutionary Computation 6 (4) (2002) 402–412.
[17] E. Angel, E. Bampis, L. Gourvès, Approximating the Pareto curve with local search for the bicriteria TSP(1,2) problem, Theoretical Computer Science 310 (1–3) (2004) 135–146.
[18] L. Paquete, T. Stützle, A two-phase local search for the biobjective traveling salesman problem, in: Evolutionary Multi-Criterion Optimization, EMO 2003, Springer, 2003, pp. 479–493.
[19] J. Knowles, D. Corne, The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation, in: IEEE Congress on Evolutionary Computation, CEC 1999, 1999, pp. 98–105.
[20] J. Bader, E. Zitzler, HypE: an algorithm for fast hypervolume-based many-objective optimization, Evolutionary Computation 19 (1) (2011) 45–76.
[21] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.
[22] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Transactions on Evolutionary Computation 13 (2) (2009) 284–302.
[23] P. Larranaga, J.A. Lozana, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer, 2001.
[24] L. Ke, Q. Zhang, R. Battit, MOEA/D-ACO: a multiobjective evolutionary algorithm using decomposition and ant colony, IEEE Transactions on Cybernetics (2013) in press.
[25] F. Gao, A. Zhou, G. Zhang, An estimation of distribution algorithm based on decomposition for multiobjective TSP, in: 8th International Conference on Natural Computation, ICNC 2012, 2012, pp. 817–821.
[26] G. Reinelt, TSPLIB—a traveling salesman problem library, ORSA Journal on Computing 3 (1991) 376–384.
[27] E. Zitzler, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.
[28] Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: a regularity model-based multiobjective estimation of distribution algorithm, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 41–63.
[29] F.Q. Gu, H.L. Liu, A novel weight design in multi-objective evolutionary algorithm, in: International Conference on Computational Intelligence and Security, 2010, pp. 137–141.