DEVELOPING A HYBRID DATA MINING APPROACH BASED ON MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION FOR SOLVING A TRAVELING SALESMAN PROBLEM

Abdorrahman Haeri[1], Reza Tavakkoli-Moghaddam[2]

Department of Industrial Engineering and Center of Excellence for Intelligence Based Experimental Mechanic, College of Engineering, University of Tehran, Tehran, Iran
E-mail: [2]tavakoli@ut.ac.ir (corresponding author)

Abstract. A traveling salesman problem (TSP) is an NP-hard optimization problem. So it is necessary to use intelligent and heuristic methods to solve such a hard problem in a less computational time. This paper proposes a novel hybrid approach, which is a data mining (DM) based on multi-objective particle swarm optimization (MOPSO), called intelligent MOPSO (IMOPSO). The first step of the proposed IMOPSO is to find efficient solutions by applying the MOPSO approach. Then, the GRI (Generalized Rule Induction) algorithm, which is a powerful association rule mining, is used for extracting rules from efficient solutions of the MOPSO approach. Afterwards, the extracted rules are applied to improve solutions of the MOPSO for large-sized problems. Our proposed approach (IMOPSP) conforms to a standard data mining framework is called CRISP-DM and is performed on five standard problems with bi-objectives. The associated results of this approach are compared with the results obtained by the MOPSO approach. The results show the superiority of the proposed IMOPSO to obtain more and better solutions in comparison to the MOPSO approach.

Keywords: traveling salesman problem, data mining, multi-objective PSO, association rule mining; CRISP-DM algorithm, GRI algorithm.

Reference to this paper should be made as follows: Haeri, A.; Tavakkoli-Moghaddam, R. 2012. Developing a hybrid data mining approach based on multi-objective particle swarm optimization for solving a traveling salesman problem, *Journal of Business Economics and Management* 13(5): 951–967.

JEL Classification: C61.

1. Introduction

A traveling salesman problem (TSP) is a traditional and well-known optimization problem in the field of operations research. There are *n* cities and distances between cities are specific and known. In this paper, a symmetric TSP is considered, in which the distance from city *i* to city *j* is equal to the distance from city *j* to city *i*. A salesman starts from one arbitrary city and visits all cities exactly once and at the end returns to the first city. In other words, the aim of a TSP is to find a tour between cities that minimizes the

total distances travelled by the salesman. This problem can be explained by graphs, in which cities are vertices of a graph and the route between two cities is an edge in the graph. The weight of each edge is the distance between two cities connected by an edge. The Hamilton tour is a tour between vertices that visits all vertices once. Therefore in this case, the purpose is to find a Hamilton tour so that the sum of edge weights in the tour is minimized. The input information is the distance matrix that shows the distance among any two pairs of cities. It can be obtained from coordination of cities in the two or three-dimensional space. Each city is specified with horizontal and vertical indices in a two-dimensional plane. The distance between each pair of two cities is equivalent to the Euclidean distance between two points in the two-dimensional space.

There are many researches in the literature that use intelligent approaches, such as artificial neural network (ANN), for solving TSPs. Masutti and de Castro (2009) developed a modified version of an immune self-organizing neural network for solving a TSP. The results show that the performance of their proposed algorithm is better than other neural network methods in the literature. Leung *et al.* (2004) applied an expanding self-organizing map, called ESOM on some examples that their range is varied from 50 to 2400 cities. The results show the superiority of the proposed approach over some of other SOM approaches in the literature. Jin *et al.* (2003) proposed an integrated SOM (ISOM) with a new learning rule that combines three learning procedures available in the literature. Yan and Zhou (2006) applied a three-tier multi-agent approach to present solutions for TSPs. These tiers are ant colony optimization agent, genetic algorithm agent, and fast local searching agent. The results of this paper indicate the suitable performance of the proposed approach for both solution quality and computational time criteria. Tan *et al.* (2006) developed an improved multi-agent approach to solve large-scale TSPs. The proposed approach uses three kinds of agents with different functions that are generating a new solution, optimizing the current solution group, and refining the best solution. The experimental results show the good performance of the proposed approach. Liu *et al.* (2006) developed a hybrid of particle swarm optimization (PSO) and memetic algorithm for solving TSPs. In addition, it includes a simulated annealing (SA) local search based approach.

In the real world, there is usually more than one objective function. For example it is necessary to minimize the distance, cost, time and risk simultaneously. So it is necessary to consider more than one distance matrices between cities to minimize multiple objectives. In this paper, like Cheng *et al.* (2011), Samanlioglu *et al.* (2008), Jozefowiez *et al.* (2008) and Zhong *et al.* (2010), a bi-objectives TSP is considered. In multi-objective problems, there is no best solution, in which a collection of solutions is considered as best solutions. This collection, which is called non-dominated (efficient) solutions, is related to the dominance concept investigated in below. Consider that A and B are two solutions in a minimization multi-objective problem. Suppose that the following two conditions are occurred.

  a) The objective values of solution A are less than or equal to the objectives of solution B.
  b) The value of at least one objective of solution A is less than the considered objective of solution B.

In this condition, it is called that solution A dominates solution B. Indeed, solution B does not have any advantage in comparison with solution A. If there is not any solution that dominates solution A, it is called that solution A is a non-dominated solution. The aim of solving multi-objective problems is to find non-dominated solutions. In the literature, it is an evident approach in the recent years for solving problems with multiple objectives. Jaszkiewicz (2002) presented a genetic local search for multi-objective optimization problems to create non-dominated solutions. In each iteration of a local search, the process is implemented on generated offspring in order to increase the quality of solutions. At the end, he examined the efficiency of the proposed approach on TSP instances. Yang *et al.* (2008) considered a dynamic multi-objective TSP (DMO-TSP) of a mobile communication network that its attributes change dynamically. Attributes are the number of cities and conflict a degree between objectives. They proposed a parallel form of multi-algorithm co-evolution strategy (MACS) for solving this complicated model.

It is obvious from the literature that multi-objective particle swarm optimization (MOP-SO) approach is not used for solving multi-objective TSPs. It is also apparent that the useful data mining (DM) approach is not used for solving TSPs effectively. It is worth noting that DM is a collection of computational techniques that can be used for finding knowledge, hidden patterns and rules from data in different sciences. Ince and Aktan (2009) introduced and applied some of data mining techniques in their research. In the recent years, data mining approach has been used for optimization purposes. In this paper, one of data mining techniques is used for extracting rules from non-dominated solutions in a multi-objective TSP (MOTSP). Indeed, this paper presents a hybrid approach consisting of the MOPSO procedure and data mining process for solving MOTSP. Whereas DM is an intelligent approach for solving problems, the proposed approach is then called intelligent MOPSO (IMOPSO). Three major steps of the proposed IMOPSO for solving MOTSPs are stated as follows:

Step 1: Solving some MOTSPs with the MOPSO approach.

Step 2: Extracting rules from non-dominated solutions that are obtained in Step 1 in order to establish a rule set for each problem.

Step 3: Using a rule base for of each problem to improve obtained solutions of larger problems.

In this paper, the single and multi-objective PSO is explained first. Afterwards, the proposed IMOPSO is considered. For this purpose, the data mining process for extracting rules from non-dominated solutions of the MOPSO is demonstrated. After that the results of the MOPSO and the proposed IMOPSO are compared. Finally, conclusion and suggestions for future researches are stated.

2. Single and multi-objective PSO

Particle swarm optimization (PSO) is a meta-heuristic method that is used efficiently for solving NP-hard problems, such as TSP in the previous studies (e.g., Ouyang, Zhou (2011)), Shen *et al.* (2006), Liu *et al.* (2006), Zhang and Si (2010) and Shi *et al.* (2004)).

This method simulates a moving group of fish or birds, called particle or swarm in PSO, respectively. In comparison with genetic algorithms (GAs), particle and swarm are similar to chromosome and population, respectively. In PSO, particles created in the first iteration are not excluded and are remained until the end. In each iteration, every particle has a position and a velocity, in which the positions of particles are updated in order to obtain better solutions. The best position for each particle is stored as personal best (i.e., *pbest*) position. The best position of all of particles is stored as global best (i.e., *gbest*) position. The best position is the position that has the minimum/maximum objective function. Symbols of the PSO procedure are stated as follows:

$x_{i,1}, x_{i,2}, \ldots, x_{i,n}$: *n* continuous decision variables

$x_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,n}]$: Position vector in the *i*-th iteration.

$v_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,n}]$: Velocity vector in the *i*-th iteration.

pbest$_i$: Vector that stores the best position of the particles during iterations

gbest$_i$: Best positions of all particles during iterations

c_1, c_2: Predefined coefficients

r_1, r_2: Random numbers between 0 and 1, generated for each particle in each iteration

w: Inertia factor that can be equal to one

The basic PSO approach for solving single-objective problems is stated as follows:

1) Initial particles are generated randomly.

2) Initial velocities of particles are zero.

3) In each iteration, the velocity of each particle is computed by:

$$v_{i+1} = w.v_i + c_1.r_1.(pbest_i - x_i) + c_2.r_2.(gbest_i - x_i). \tag{1}$$

4) The position of each particle is updated by using the following equation.

$$x_{i+1} = x_i + v_{i+1.} \tag{2}$$

5) The above process is repeated until a termination condition is occurred. This condition is usually a number of iterations.

PSO is suitable for continuous variables. It is worth noting that a TSP is a problem with integer variables. So it is necessary to modify PSO to be applicable for solving TSPs. For this purpose, the rank ordered value (ROV) method is used as same as given in Liu *et al.* (2006). For solving an *n*-city TSP, a string with *n* numbers is defined, namely original string. Numbers of this string are in [0, 1] range, in which each number is correspondent to one city. Corresponding to each original string, a tour consisting of *n* cities is defined by using the ROV method. This method performs in three steps explained by:

1) Sorting the numbers of the original string in an ascending order.
2) Specifying the rank of each numbers in an ascending order.
3) Creating a tour with the rank of cities in an ascending order.

For example, consider a TSP with five cities and assume that the original string is A = [0.23, 0.11, 0.58, 0.49, 0.87]. The sorted order of A is as follows: A = [0.11, 0.23, 0.49, 0.58, 0.87]. So the corresponding tour with A is [2, 1, 4, 3, 5].

In this paper, a multi-objective particle swarm optimization (MOPSO) algorithm is used for solving a multi-objective traveling salesman problem (MOTSP). For this purpose, a crowding distance (CD) factor is defined on the basis of the concept given in Deb *et al.* (2002). This factor is used for specifying how much a solution is crowded with other solutions. In other words, it is a density estimator used for non-dominated solutions. Consider a collection that includes *m* non-dominated solutions. The CD factor for each solution is calculated by the following steps:

1) For each objective, sort solutions in an ascending order.

2) The CD for the first and last solutions in order is equal to ∞. In an applicable case, it can be equal to a big number.

3) For the other solutions, the CD is calculated by the relation shown below:

$$CD_i = (f_{i+1} \, f_{i-1} \,)/(f_{max} - f_{min}). \tag{3}$$

Symbols of Eq. (3) are defined as follows:

CD_i: Crowding distance (CD) factor of the *i*-th solution in the sorted collection of the non-dominated solutions.

f_{i+1}: Objective function value of the (*i*+1)-th solution in the sorted collection of the non-dominated solutions.

f_{i-1}: Objective function value of the (*i*-1)-th solution in the sorted collection of the non-dominated solutions.

f_{max}: Maximum objective function value in the sorted collection of the non-dominated solutions.

f_{min}: Minimum objective function value in the sorted collection of the non-dominated solutions.

4) The overall CD factor for each solution is the sum of CD factors for each objective function.

Steps of MOPSO are explained as follows:

1) Initial particles are generated randomly.

2) Initial velocities of particles are zero.

3) Evaluate all particles and select non-dominated solutions from swarm. Non-dominated solutions are stored in a pool, called repository. In each iteration, new non-dominated solutions are added to repository. If any of the current solutions of repository is dominated by new solutions, it is deleted from repository. The capacity of repository is limited and is defined by the user. Suppose that a number of non-dominated solutions are more than the capacity of repository. So it is necessary to delete (or exclude) some solutions from repository. In this situation, non-dominated solutions are sorted in an ascending order on the basis of their CD factor. Solutions with the smaller CD factor are excluded. It means that solutions, which are more crowded with other solutions, are deleted. It results in that solutions, which are less crowded with other solutions, are remained. It results in more diversification in the space search process during the algorithm implementation.

4) *pbest* of each particle is updated. In the first iteration, *pbest* is equal to the initial position of a particle. In the next iterations, *pbest* for each particle is updated by using three simple rules as follows:

   a) If the current solution (i.e., position) dominates the *pbest* solution, the *pbest* solution is equal to the current solution.
   b) If the *pbest* solution (i.e., position) dominates the current solution, the *pbest* solution is remained without any change.
   c) If neither of them dominates the other, one of them is selected randomly as the *pbest* solution.

5) In each iteration, the velocity of each particle is calculated by:

$$v_{i+1} = w.v_i + c_1.r_1.(pbest_i - x_i) + c_2.r_2.(rep_H - x_i). \tag{4}$$

There is a main difference in the velocity equation between single and multi-objective problems. In multi-objective problems, there is not any global solution. Instead, there is a repository of non-dominated solutions. H implies to one of the solutions that is selected from repository. There are some ways for selecting a solution from repository at random. In this paper, similar to Tsou *et al.* (2007) it is selected from less crowded solutions. For this purpose, solutions in repository are sorted on the basis of their CD factors. Then 10% of solutions with less CD factors are specified and H is selected from them randomly. So, rep_H is a vector stating position of the selected solution and is used in Eq. (4).

6) Update the position of each particle by using the following equation.

$$x_{i+1} = x_i + v_{i+1}. \tag{5}$$

7) The above process is repeated until a termination condition is occurred. This condition is usually a number of iterations.

It is necessary to tune up some parameters before running the algorithm. It is recommended that number of particles is set between 20 and 80 and number of iterations (swarms) is set between 80 and 120 (Coello, Lechunga 2002). In this paper, we consider 20 for the swarm size and 80 for a number of iterations. c_1 and c_2 coefficients are equal to 2. The repository capacity of should be defined by the user. In this paper, we consider that the repository capacity is equal to 20.

3. Data mining process

The data mining process is expressed on the basis of a standard procedure that is called the CRISP-DM algorithm and is explained in the previous studies, such as Olson and Delen (2008), Nisbet *et al.* (2009), Mladenić (2003), Han and Kamber (2006), Gupta (2006), Lin *et al.* (2008), Maimon and Rokach (2005), Riccia (2000) and Larose (2006). The six steps of this algorithm are as follows:

3.1. Business understanding

In this phase, the objective of the data mining process is defined. Usually, the business objective is considered in data mining studies. So this phase is named business

understanding. However, the objective in this study is to find rules in non-dominated solutions for some examples of TSPs. In other words, in this paper the goal of the data mining study is finding suitable rules and patterns in non-dominated solutions of a TSP.

3.2. Data understanding

In this phase, a perception from data set is obtained. There are two cost matrixes between cities in each TSP problem. Usually the cost matrix is the distance matrix between cities. So there are two distance matrices that show distances between cities. In an n-city TSP, distance matrices are $n \times n$ matrices as shown by $D_{n,1}$ and $D_{n,2}$. In a bi-objectives TSP, each edge that connects two cities has two weights related to the two distance matrices. For example, consider p-q edge that connects p and q cities. S_p is the set of edges that connect p to other cities. $Sum_{p,1}$ is the sum of the weight of edges in S_p on the basis of the $D_{n,1}$ matrix. Similarly $Sum_{p,2}$ is the sum of the weight of edges in S_p on the basis of the $D_{n,2}$ matrix.

The considered data set includes a table consisting of some rows and some columns. Rows and columns of this table are called records and fields, respectively. Each record presents one edge between two cities. Each field presents one of edge attributes. In this paper, nine attributes are considered for edges that are mentioned in below:

A field: This field is a binary (0 or 1) field that relates to existence of an edge in a non-dominated solution. If an edge exists in a non-dominated solution the value of this field is equal to 1. Also if an edge does not exist in a non-dominated solution the value of this field is equal to 0. As before said the goal of data mining process is finding rules in non-dominated solutions. So it is necessary to focus on edges that exist in non-dominated solutions. Therefore in data mining process, only edges with A = 1, which exist in a non dominated solution, are important. In other words, DM analysis is performed on edges with A = 1.

F1 field: The normalized value of each edge weight related to the whole matrix in the objective function 1.

F2 field: The normalized value of each edge weight related to the whole matrix in the objective function 2.

G1 field: The normalized value of each edge weight in S_p set in the objective function 1.

G2 field: The normalized value of each edge weight in S_p set in the objective function 2.

In this paper, the max-min method is used for normalizing. For example, consider a set of n variables that are called $x_1, x_2, \dots , x_n$. The normalized values of this set are shown by $nx_1, nx_2, \dots , nx_n$. $min - x$ and max-x are the minimum and maximum values of the set, respectively. Each value is normalized by using Eq. (6) stated as follows:

$$nx_i = (x_i - min - x)/(max - x - min - x). \tag{6}$$

3.3. Data preparation

Usually a pure data set is not suitable for performing data mining algorithms. Data preparation provides the possibility to present a standard framework for decision making and

comparison. Since data in different problems have different values, so it is necessary to convert values to standard and normalized values. Thus in this phase, preparation of data set is performed. The value of field A is equal to 0 or 1 and values of F1, F2, G1, and G2 fields are in [0, 1] range. So there is no need for changing these values any more.

3.4. Modeling

In this phase, a suitable data mining algorithm is performed on the normalized data set, and then the results are obtained. Association rule mining (ARM) algorithms apply on a data set of records and fields to efficiently extract suitable rules that explain relationships between fields. In general for applying ARM algorithms, fields are specified as input or output fields (factors). The ARM algorithms present if-then rules to explain relations between input and output fields (factors). For example, consider this rule: If "B < x" then "A = 1". Antecedent of the rule is "B < x" and B is one of input fields. "A = 1" is consequent of the rule and A is one of output fields. Rules have two major indices, called support and confidence. For aforesaid simple rule support is the percentage of records that "B < x" condition is occurred. This value is shown with Y. Consider percentage of records that both of "B < x" and "A = 1" conditions are occurred. This value is shown with X. Confidence is equal to division of X to the Y. Indeed confidence is the accuracy of rule and is a good measure for specifying how much a rule is reliable. Support and confidence are two important criteria for selecting suitable and efficient rules. Rules with high support are frequent and rules with high confidence have high accuracy.

The goal of this study is to find suitable rules about edges in non-dominated solutions. In other words, the purpose is to specify which edges have more chance to be in a non-dominated solution. So output field (goal field) is existence of an edge in a non-dominated solution. In this case, field A is output field and other fields that are attributes of edges (i.e., F1, F2, G1 and G2) are input fields. There are two major ARM algorithms, namely Apriori and GRI (Generalized Rule Induction). It is necessary to mention that the Apriori algorithm does not accept continuous fields. Since that input fields include continuous values, this algorithm is not applicable for the considered data set, and the GRI algorithm is used for extracting rules. The GRI method is introduced and applied in previous studies, such as Larose (2005), Abbas *et al.* (2002) and Bramer (2007, 1999). To perform the GRI algorithm, the SPSS Clementine 11.1 software is used.

3.5. Evaluation

In this phase, the results of the previous phase are evaluated and analyzed. For each non-dominated solution, rules that include "A = 1" term as a consequent are considered. Since that number of rules is much, it is necessary to select some of rules for more analysis. In this paper, since confidence is the good criterion for rule selection, a threshold for rule confidence is defined. Rules that their confidence is lower than threshold are excluded and rules that their confidence is higher than threshold are stored for more analysis in the next step. Threshold selection for confidence of rules is user-defined and depends to the nature of the considered problem and its needed accuracy. In this paper, 70% is a good threshold that can satisfy needed accuracy and is considered as the minimum threshold for confidence of rules.

3.6. Deployment

In this phase, the results of the previous steps (i.e., extracted rules) are used for solving MOTSP problems. Indeed, the set of extracted rules from efficient solutions of an MOTSP problem is considered. For example, consider RS_n that is a rule set of efficient solutions from an n-city bi-objectives problem. This rule set contains m rules as mentioned in Table 1.

Table 1. Rules that constitute the RS_n rule set

Number of rule	Antecedent	Consequent	Confidence
1	A_1	$A = 1$	CR_1
2	A_2	$A = 1$	CR_2
…	…	…	…
m	A_m	$A = 1$	CR_m

After establishing a rule set, it is necessary to use of the rule set to solve another k-city ($k > n$) problem. The following steps that explain the IMOPSO approach are performed for this purpose:

1) First a k-city bi-objectives TSP problem with the MOPSO method is solved. The obtained efficient solutions constitute a set that is called ES_k. Suppose that ES_k contains p efficient solutions.

2) For i-th ($1 \leq i \leq p$) solution of the ES_k set, consider the j-th ($1 \leq j \leq m$) rule of RS_n. If consequent of the j-th rule is "False", then the next rule will be selected. If consequent of the j-th rule is "True", it means that this rule states the conditions that two cities (e.g., x and y) with probability CR_j can be adjacent.

3) Therefore, it is probable that cities x and y are adjacent in efficient solutions of the k-city bi-objective TSP problem. If cities x and y are adjacent in the i-th solution, it will be remained without any change.

4) If cities x and y are not adjacent in the i-th solution, a random number (e.g., RI) will be generated and compared with the j-th rule confidence. If ($RI \leq CRj$), then the i-th solution will be changed so that cities x and y will be adjacent. For this purpose, a tour of cities will be considered and one of the adjacent cities of x (e.g., z) will be selected. Then, position of y and z will be exchanged to reach a new tour that includes x and y as adjacent cities.

5) The previous steps (i.e., Steps 2, 3 and 4) are performed several times to obtain a diverse set of solutions.

6) At the end, new set of solutions are explored to select efficient solutions from that. Afterwards, the new obtained efficient solutions from the IMOPSO approach are compared with the previous efficient solutions obtained by the MOPSO approach in order to find the final set of efficient solutions. This comparison specifies that how the proposed hybrid approach can improve the ability to reach the efficient solutions.

4. Experimental results and discussion

The proposed approach is performed on five examples of TSPs. Test problems are taken from the TSPLIB (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html), whose data set includes the TSP data set with a single objective. However, in this paper, two objectives TSP examples are needed. To prepare these examples, a trick similar to Jaszkiewicz (2002) is used. For each *n*-city problem, two *n*-city examples are considered. So the distance matrices from a bi-objectives problem are taken from two problems with a single distance matrix. For example, to create a 29-city bi-objectives TSP, "bayg29" and "bays29" are considered and each distance matrix is taken from one of them. The obtained TSP with two distance matrices is called EX29, in which EX29 is a bi-objectives TSP problem that includes both "bayg29" and "bays29" distance matrices. Other bi-objectives examples and their correspondent single-objective problems are stated in Table 2.

Table 2. Original examples from the TSPLIB corresponding bi-objectives examples

Original example	New two objectives example
bayg29, bays29	EX29
dantzig42, swiss42	EX42
gr48, hk48	EX48
eil76, pr76	EX76
kroD100, kroE100	EX100

For each example, the MOPSO approach is used to reach efficient solutions (ESs). Then, the GRI algorithm is applied to extract rules from efficient solutions. Each rule is an "If-Then" statement that includes some terms as antecedents and consequent. In this paper, F1, F2, G1 and G2 fields are antecedents and field A is consequent. For example, one rule that is extracted from EX29 is stated as follows:

If ((F1 < 0.17487) & (F2 > 0.146366) & (G1 < 0.222709) & (0.212743< G2 < 0.292752)), then A = 1

All rule sets related to EX29, EX42, EX48, and EX76 are stated in Appendix. There are some similarities between rules such as:
  • Consequent of all rules is "A = 1"
  • Antecedents of all rules are about F1, F2, G1 and G2 fields.

Differences among rules are stated as follows:
  • Some rules do not include all fields. For example, this is a rule from EX42 that does not include F2 and G1 fields.
    If (F1 < 0.356874) and (0.247236 < G2 < 0.24897), then A = 1.
  • Determined ranges of F1, F2, G1 and G2 fields are different numerical values in different rules.

Extracted rules are used to improve efficient solutions from the MOPSO approach for larger MOTSP problems. For example, a rule set from EX29, called EX29 rules set, is

used to improve efficient solutions of EX42, EX48, EX76 and EX100. Table 3 shows the results of applying the EX29 rule set to solve large-scale problems. In addition, Table 3 includes comparison between the performance of MOPSO and intelligent MOPSO (IMOPSO) approaches. The first column of this table shows the considered problem. The second and third columns include the number of efficient solutions (ES) obtained by the applying MOPSO and IMOPSO approaches, respectively. The fourth column shows the number of efficient solutions of the MOPSO approach (MOPSO-ES) that are dominated by the efficient solutions of the IMOPSO approach (IMOPSO-ES). The fifth column shows the number of the efficient solutions of the IMOPSO method (IMOPSO-ES) that are dominated by the efficient solutions of the MOPSO approach (MOPSO-ES). The sixth column states the total number of a collection that includes MOPSO-ES and IMOPSO-ES simultaneously. The seventh and eighth columns contain the number of MOPSO-ES and IMOPSO-ES in the new collection, respectively.

Table 3. Results of applying the EX29 rule set

Problem	No. of ES for MOPSO approach	No. of ES for IMOPSO approach	No. of MOPSO-ES dominated by IMOPSO-ES	No. of IMOPSO-ES dominated by MOPSO-ES	Total number of Collection of MOPSO-ES and IMOPSO-ES	Number of MOPSO-ES in collection	Number of IMOPSO-ES in collection
EX42	11	7	7 (64%)	0 (0%)	$(11-7) + (7-0) = 11$	$11 - 7 = 4$	$7 - 0 = 7$
EX48	10	9	10 (100%)	0 (0%)	$(10-10) + (9-0) = 9$	$10 - 10 = 0$	$9 - 0 = 9$
EX76	11	8	11 (100%)	0 (0%)	$(11-11) + (8-0) = 8$	$11 - 11 = 0$	$8 - 0 = 8$
EX100	7	13	7 (100%)	0 (0%)	$(7-7) + (13-0) = 13$	$7 - 7 = 0$	$13 - 0 = 13$
Average			91%	0%			

The fourth column of Table 3 shows that efficient solutions of the IMOPSO approach dominated all of the efficient solutions of the MOPSO approach for EX48, EX76 and EX100. Only four MOPSO solutions for EX42 are not dominated by the IMOPSO efficient solutions. The fifth column of Table 3 indicates that there is no efficient solution of the IMOPSO approach that is dominated by any solution of the MOPSO approach for all considered problems (EX42, EX48, EX76 and EX100). The seventh and eighth columns of Table 3 show that for EX48, EX76 and EX100 all solutions of collection of MOPSO-ES and IMOPSO-ES are dedicated to the IMOPSO-ES. Only four solutions of MOPSO-ES in EX42 exist in collection of MOPSO-ES and IMOPSO-ES.

Similarly to Table 3, Tables 4 to 6 show the results of applying EX42, EX48 and EX76 rule sets, respectively. Table 4 shows that efficient solutions of the IMOPSO approach

dominate all efficient solutions of the MOPSO approach for EX76 and EX100. Only three MOPSO solutions of EX48 are not dominated by the IMOPSO solutions. Moreover, there is no efficient solution of the IMOPSO approach that is dominated by any solution of the MOPSO approach for all problems (EX48, EX76 and EX100).

Table 5 shows that efficient solutions of the IMOPSO approach dominate all efficient solutions of the MOPSO approach for EX76. Two MOPSO solutions of EX100 are dominated by the IMOPSO solutions. Moreover, there is no solution of the IMOPSO approach that is dominated by any solution of the MOPSO approach for EX76. Also only three solutions of the IMOPSO approach are dominated by solutions of the MOPSO approach for EX100. Table 6 shows that efficient solutions of the IMOPSO approach dominate three solutions of the MOPSO approach for EX100. Also two solutions of the IMOPSO approach are dominated by solutions of MOPSO for EX100.

Table 4. Results of applying the EX42 rule set

Problem	No. of ES for MOPSO approach	No. of ES for IMOPSO approach	No. of MOPSO-ES dominated by IMOPSO-ES	No. of IMOPSO-ES dominated by MOPSO-ES	Total number of Collection of MOPSO-ES and IMOPSO-ES	Number of MOPSO-ES in collection	Number of IMOPSO-ES in collection
EX48	10	6	7 (70%)	0 (0%)	(10 − 7) + (6 − 0) = 9	10 − 7 = 3	6 − 0 = 6
EX76	11	14	11 (100%)	0 (0%)	(11 − 11) + (14 − 0) = 14	11 − 11 = 0	14 − 0 = 14
EX100	7	5	7 (100%)	0 (0%)	(7 − 7) + (5 − 0) = 5	7 − 7 = 0	5 − 0 = 5
Average			90%	0%			

Table 5. Results of applying the EX48 rule set

Problem	No. of ES for MOPSO approach	No. of ES for IMOPSO approach	No. of MOPSO-ES dominated by IMOPSO-ES	No. of IMOPSO-ES dominated by MOPSO-ES	Total number of Collection of MOPSO-ES and IMOPSO-ES	Number of MOPSO-ES in collection	Number of IMOPSO-ES in collection
EX76	11	11	11 (100%)	0 (0%)	(11 − 11) + (11 − 0) = 11	11 − 11 = 0	11 − 0 = 11
EX100	7	13	2 (29%)	3 (23%)	(7 − 2) + (13 − 3) = 15	7 − 2 = 5	13 − 3 = 10
Average			64.5%	11.5%			

Tables 3 to 6 show the absolute superiority of the IMOPSO approach in comparison to the MOPSO approach in the following there aspects:

a) Most of efficient solutions of the MOPSO approach are dominated by the efficient solutions of the IMOPSO approach.
b) Most of efficient solutions of the IMOPSO approach are not dominated by the efficient solutions of the IMOPSO approach.
c) Applying IMOPSO approach results in presenting new efficient solutions in comparison to the MOPSO approach.

Finally, Table 7 shows a summary of results for applying the EX29, EX42, EX48 and EX76 rule sets. In this table, the average percentage of efficient solutions of the MOPSO approach dominated by the efficient solutions of IMOPSO approach is presented. In addition, this table includes the average percentage of efficient solutions of the IMOPSO approach that are dominated by the efficient solutions of the MOPSO approach. At whole, this indicates that most solutions of the MOPSO approach are dominated by the efficient solutions of the IMOPSO approach. Instead, few solutions of the IMOPSO approach are dominated by the efficient solutions of the MOPSO approach.

Table 6. Results of applying the EX76 rule set

Problem	No. of ES for MOPSO approach	No. of ES for IMOPSO approach	No. of MOPSO-ES dominated by IMOPSO-ES	No. of IMOPSO-ES dominated by MOPSO-ES	Total number of Collection of MOPSO-ES and IMOPSO-ES	Number of MOPSO-ES in collection	Number of IMOPSO-ES in collection
EX100	7	13	3 (43%)	2 (15%)	$(7-3) + (13-2) = 15$	$7-3=4$	$13-2=11$
Average			(43%)	(15%)			

Table 7. Summary of results for applying the EX42, EX48, EX76 and EX100 rule sets

Problem	Average percentage of MOPSO-ES that are dominated by the IMOPSO-ES	Average percentage of IMOPSO-ES that are dominated by the MOPSO-ES
EX29	91%	0%
EX42	90%	0%
EX48	64.5%	11.5%
EX76	43%	15%

5. Conclusions

This paper has proposed an integrated intelligent approach for solving a multi-objective traveling salesman problem (MOTSP). This approach has used data mining and multi-objective particle swarm optimization (MOPSO). First, five problems were solved by the MOPSO approach. Then, data mining (DM) was used to find knowledge from ef-

ficient solutions of MOTSPs. So DM based on MOPSO was called intelligent MOPSO (IMOPSO) as a novel hybrid approach. Then, the GRI algorithm, which was an association rule mining algorithm, was performed and the extracted knowledge was explained as if-then rules. Extracted rules were used for solving new problems. The process of rule extracting and applying them to improve solutions of the MOPSO approach was stated in a standard data mining framework, called CRISP-DM algorithm. The proposed approach was compared with the MOPSO approach resulting that the IMOPSO approach has two major benefits. First, it produces new efficient solutions and therefore increases the number of non-dominated (efficient) solutions. The second benefit is that most solutions of the MOPSO approach are dominated by the efficient solutions of the IMOPSO approach. So, the IMOPSO approach presents better solutions. Indeed, In addition, a few solutions of the IMOPSO approach were dominated by solutions of the MOPSO approach. In other words, the IMOPSO approach produced solutions that were better than solutions of the MOPSO approach in terms of the solution quality and quantity. Table 7 shows that 91, 90, 64.5 and 43% of efficient solutions of the MOPSO approach are dominated by the efficient solutions of the IMOPSO approach in case of applying EX29, EX42, EX48 and EX76 rule sets, respectively. In addition, it has shown that only 0, 0, 11.5 and 15% of efficient solutions of the IMOPSO approach are dominated by the efficient solutions of the MOPSO approach in case of applying EX29, EX42, EX48 and EX76 rule sets, respectively. Furthermore, in multi-objective problems, finding many numbers of efficient solutions is a major benefit. The IMOPSO approach provides more efficient solutions in comparison to the MOPSO approach. Applying the hybrid proposed approach in this paper to the other optimization problems can be suggested for future research. Furthermore, it is suggested to develop a rule-based optimization approach that uses other rule extracting techniques during the optimization process.

References

Abbass, H. A.; Sarker, R. A.; Newton, C. S. 2002. *Data Mining: a Heuristic Approach*. PA: Idea Group Publishing.

Bramer, M. A. 1999. *Knowledge Discovery and Data Mining*. London: The Institution of Electrical Engineers.

Bramer, M. A. 2007. *Principles of Data Mining*. London: Springer.

Coello, C. A.; Lechunga, M. S. 2002. MOPSO: a proposal for multiple objective particle swarm optimization, in *Proc. of the IEEE World Congress on Evolutionary Computation*. Hawaii, 1051–1056.

Cheng, J.; Zhang, G.; Li, Z.; Li, Y. 2012. Multi-objective ant colony optimization based on decomposition for bi-objective traveling salesman problems, *Soft Computing* 16(4): 597–614.

Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. 2002. A fast and elitist multi-objective genetic algorithm: NSGAII, *IEEE Transaction on Evolutionary Computation* 6(2): 181–197. http://dx.doi.org/10.1109/4235.996017

Gupta, G. K. 2006. *Introduction to Data Mining with Case Studies*. New Delhi: Prentice-Hall of India.

Han, J.; Kamber, M. 2006. *Data Mining: Concepts and Techniques*. USA: Elsevier Inc. Available from Internet: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html

Ince, H.; Aktan, B. 2009. A comparison of data mining techniques for credit scoring in banking: a managerial perspective, *Journal of Business Economics and Management* 10(3): 233–240. http://dx.doi.org/10.3846/1611-1699.2009.10.233-240

Jaszkiewicz, A. 2002. Genetic local search for multi-objective combinatorial optimization, *European Journal of Operational Research* 137(1): 50–71.
http://dx.doi.org/10.1016/S0377-2217(01)00104-7

Jin, H.-D.; Leung, K.-S.; Wong, M.-L.; Xu, Z.-B. 2003. An efficient self-organizing map designed by genetic algorithms for the traveling salesman problem, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 33(6): 877–888.
http://dx.doi.org/10.1109/TSMCB.2002.804367

Jozefowiez, N.; Glover, F.; Laguna, M. 2008. Multi-objective meta-heuristics for the traveling salesman problem with profits, *Journal of Mathematical Modelling and Algorithms* 7(2): 177–195. http://dx.doi.org/10.1007/s10852-008-9080-2

Larose, D. T. 2005. *Discovering Knowledge in Data: an Introduction to Data Mining*. NJ: John Wiley & Sons, Inc.

Larose, D. T. 2006. *Data Mining Methods and Models*. NJ: John Wiley & Sons, Inc.

Leung, K.-S.; Jin, H.-D.; Xu, Z.-B. 2004. An expanding self-organizing neural network for the traveling salesman problem, *Neurocomputing* 62(1–4): 267–292.
http://dx.doi.org/10.1016/j.neucom.2004.02.006

Lin, T. Y.; Xie, Y.; Wasilewska, A. 2008. *Data Mining: Foundations and Practice*. Heidelberg: Springer. http://dx.doi.org/10.1007/978-3-540-78488-3

Liu, B.; Wang, L.; Jin, Y.-H.; Huang, D.-X. 2006. An effective PSO-based memetic algorithm for TSP, *Lecture Notes in Control and Information Sciences* 345: 1151–1156.
http://dx.doi.org/10.1007/978-3-540-37258-5_149

Maimon, O. Z.; Rokach, L. 2005. *Data Mining and Knowledge Discovery Handbook*. USA: Springer. http://dx.doi.org/10.1007/b107408

Masutti, T. A. S.; de Castro, L. N. 2009. A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem, *Information Sciences* 179(10): 1454–1468. http://dx.doi.org/10.1016/j.ins.2008.12.016

Mladenić, D. 2003. *Data Mining and Decision Support: Integration and Collaboration*. USA: Kluwer Academic Publishers.

Nisbet, R.; Elder, J.; Elder, J. F.; Miner, G. 2009. *Handbook of Statistical Analysis and Data Mining Applications*. Canada: Elsevier Inc.

Olson, D. L.; Delen, D. 2008. *Advanced Data Mining Techniques*. Heidelberg: Springer.

Ouyang, A.; Zhou, Y. 2011. An improved PSO-ACO algorithm for solving large-scale TSP, *Advanced Materials Research* 143–144: 1154–1158.
http://dx.doi.org/10.4028/www.scientific.net/AMR.143-144.1154

Riccia, G. D.; Kruse, R.; Lenz, H.-J. 2000. *Computational Intelligence in Data Mining*. Italy: Springer.

Samanlioglu, F.; Ferrell, Jr. W. G.; Kurz, M. E. 2008. A memetic random-key genetic algorithm for a symmetric multi-objective traveling salesman problem, *Computers and Industrial Engineering* 55(2): 439–449. http://dx.doi.org/10.1016/j.cie.2008.01.005

Shen, B.; Yao, M.; Yi, W. 2006. Heuristic information based improved fuzzy discrete PSO method for solving TSP, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4099: 859–863.

Shi, X. H.; Xing, X. L.; Wang, Q. X.; Zhang, L. H.; Yang, X. W.; Zhou, C. G.; Liang, Y. C. 2004. A discrete PSO method for generalized TSP problem, in *Proceedings of 3rd International Conference on Machine Learning and Cybernetics*, *August 26–29, 2004*. Shanghai, China, 2378–2383.

Tan, Y.-A.; Zhang, X.-H.; Xing, L.-N.; Zhang, X.-L.; Wang, S.-W. 2006. An improved multi-agent approach for solving large traveling salesman problem, *Lecture Notes in Artificial Intelligence* 4088: 351–361.

Tsou, C.-S.; Chang, S.-C.; Lai, P.-W. 2007. *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*. Chapter five. I-Tech Education and Publishing.

Yan, S.-L.; Zhou, K.-F. 2006. Three-tier multi-agent approach for solving traveling salesman problem, *Lecture Notes in Artificial Intelligence* 4099: 813–817.

Yang, M.; Kang, Z.; Kang, L. 2008. A parallel multi-algorithm solver for dynamic multi-objective TSP (DMO-TSP), *Lecture Notes in Artificial Intelligence* 5227: 164–173.

Zhang, J.-W.; Si, W.-J. 2010. Improved Enhanced Self-Tentative PSO algorithm for TSP, in *Proceedings of 6th International Conference on Natural Computation*, *August 10–12, 2010*. Yantai, China, 2638–2641. http://dx.doi.org/10.1109/ICNC.2010.5583011

Zhong, H.; Zhuang, N.; Cai, W.; Zhang, G.; Wu, Z. 2010. Multi-objective two-depot traveling salesman problem with uncertain fleet size for hazardous materials, in *Proceedings of 8th International Conference on Supply Chain Management and Information Systems: Logistics Systems and Engineering*, *October 6–8, 2010*. Hong Kong, China, 1–8.

APPENDIX

Rule sets of EX29, EX42, EX48, and EX76

Problem	Antecedent 1 (F1 Field)	Antecedent 2 (F2 Field)	Antecedent 3 (G1 Field)	Antecedent 4 (G2 Field)	Consequent (A Field)	Confidence (%)
Ex29	F1 < 0.167098	F2 > 0.146366	G1 < 0.213662	0.212743 < G2 < 0.292752	A = 1	100
Ex29	F1 < 0.185233	F2 > 0.200393		0.215393 < G2 < 0.292752	A = 1	100
Ex29	F1 < 0.090673	0.027505 < F2 < 0.085461	G1 > 0.109936	G2 > 0.095274	A = 1	100
Ex29	F1 < 0.17487	F2 > 0.146366	G1 < 0.222709	0.212743 < G2 < 0.292752	A = 1	85.71
Ex29	0.156736 < F1 < 0.167098	F2 > 0.081533	G1 < 0.213662	0.176741 < G2 < 0.292752	A = 1	83.33
Ex29	F1 < 0.123057	0.083497 < F2 < 0.167977	G1 > 0.133524	G2 > 0.12297	A = 1	80
Ex29	0.156736 < F1 < 0.167098	F2 > 0.11886	G1 < 0.213662	0.107629 < G2 < 0.292752	A = 1	75
Ex29	F1 < 0.123057	F2 < 0.085461	G1 < 0.150334	G2 > 0.095274	A = 1	71.43
Ex29	F1 < 0.167098	F2 > 0.146366	G1 < 0.213662	0.176741 < G2 < 0.292752	A = 1	70
EX42	0.194803 < F1 < 0.24207	F2 > 0.119195	0.138495 < G1 < 0.355553	G2 < 0.182414	A = 1	100
EX42			G1 < 0.494339	0.247236 < G2 < 0.24897	A = 1	100
EX42	0.194803 < F1 < 0.195786	0.100619 < F2 < 0.168731	0.138495 < G1 < 0.494339	G2 < 0.24897	A = 1	100
EX42	0.300163 < F1 < 0.356874	0.140867 < F2 < 0.174923	0.41657 < G1 < 0.517352	G2 > 0.237426	A = 1	83.33
EX42	0.194803 < F1 < 0.24207		0.297633 < G1 < 0.355553	G2 < 0.182414	A = 1	80
EX42	F1 < 0.356874			0.247236 < G2 < 0.24897	A = 1	75
EX42	0.101025 < F1 < 0.233493	0.106811 < F2 < 0.147059	G1 < 0.494339	G2 < 0.181245	A = 1	71.43
EX48	0.167129 < F1 < 0.199907		0.25849 < G1 < 0.288211	G2 < 0.309073	A = 1	100
EX48	0.1759 < F1 < 0.199907		0.261527 < G1 < 0.288211	0.217505 < G2 < 0.512937	A = 1	100
EX48			G1 < 0.303016	0.502909 < G2 < 0.506344	A = 1	100

End of Appendix

Problem	Antecedent 1 (F1 Field)	Antecedent 2 (F2 Field)	Antecedent 3 (G1 Field)	Antecedent 4 (G2 Field)	Consequent (A Field)	Confidence (%)
EX48	F1 > 0.158357	F2 < 0.071507	0.180723 < G1 < 0.313554	G2 < 0.512937	A = 1	100
EX48	0.050323 < F1 < 0.199907	F2 < 0.162582	G1 < 0.190416	0.192244 < G2 < 0.512937	A = 1	90.91
EX48	0.070637 < F1 < 0.199907	F2 < 0.162582	G1 < 0.190416	0.192244 < G2 < 0.512937	A = 1	90
EX48	0.174053 < F1 < 0.199907		0.261527 < G1 < 0.288211	0.217505 < G2 < 0.512937	A = 1	88.89
EX48		F2 < 0.071507	0.228761 < G1 < 0.345719	G2 < 0.512937	A = 1	87.5
EX48	F1 > 0.158357	F2 < 0.071507	0.180723 < G1 < 0.345719	G2 < 0.512937	A = 1	85.71
EX48	0.146352 < F1 < 0.165743	F2 > 0.207023	0.194593 < G1 < 0.224968	G2 > 0.373047	A = 1	83.33
EX48	0.150046 < F1 < 0.19252		0.113653 < G1 < 0.277044	0.437615 < G2 < 0.512937	A = 1	81.82
EX48	F1 < 0.199907	F2 < 0.360827	0.204753 < G1 < 0.345719	0.405038 < G2 < 0.512937	A = 1	80
EX48	0.151893 < F1 < 0.165743	0.273043 < F2 < 0.580285	0.194593 < G1 < 0.242592		A = 1	78.57
EX48			0.180723 < G1 < 0.345719	0.502909 < G2 < 0.512937	A = 1	77.78
EX48	0.149122 < F1 < 0.182825	F2 > 0.207023	0.180723 < G1 < 0.213478	G2 < 0.69799	A = 1	76.92
EX48	0.150046 < F1 < 0.191597		0.113653 < G1 < 0.345719	0.436966 < G2 < 0.512937	A = 1	75
EX48	0.146352 < F1 < 0.165743	F2 > 0.207023	0.194593 < G1 < 0.224968		A = 1	73.33
EX48	F1 > 0.167129		0.25849 < G1 < 0.277044	G2 < 0.309073	A = 1	72.73
EX48			G1 < 0.345719	0.501403 < G2 < 0.512937	A = 1	71.43
EX48	0.151893 < F1 < 0.19252	0.207023 < F2 < 0.580285	G1 < 0.242132	0.423106 < G2 < 0.69799	A = 1	70.59
EX76	0.227084 < F1 < 0.234679	F2 < 0.190058	0.277293 < G1 < 0.332044	G2 > 0.160957	A = 1	100

Abdorrahman HAERI is a PhD student in Department of Industrial Engineering at University of Tehran. His interested research area is about data mining and soft computing methods, such as genetic algorithm and particle swarm optimization. His previous papers were about applying data mining methods, such as association rule mining, on optimization and decision-making problems.

Reza TAVAKKOLI-MOGHADDAM is a professor of Industrial Engineering at College of Engineering, University of Tehran in Iran. He obtained his PhD in Industrial Engineering from the Swinburne University of Technology in Melbourne (1998), his M.Sc. in Industrial Engineering from the University of Melbourne in Melbourne (1994) and his B.Sc. in Industrial Engineering from the Iran University of Science & Technology in Tehran (1989). He serves as Editorial Boards of the International Journal of Engineering and Iranian Journal of Operations Research. Additionally, he is the Executive Member of Board of Iranian Operations Research Society. He is the recipient of the 2009 and 2011 Distinguished Researcher Award and the 2010 Distinguished Applied Research Award at the University of Tehran, Iran. He has been selected as National Iranian Distinguished Researcher for 2008 and 2010. Professor Tavakkoli-Moghaddam has published 3 books, 12 book chapters, more than 450 papers in reputable academic journals and conferences.