

# Ant Colony Optimization for Multi-objective Digital Convergent Product Network

Reza Hassanzadeh<sup>a,\*</sup>, Iraj Mahdavi<sup>b</sup>, Nezam Mahdavi-Amiri<sup>c</sup>

<sup>a</sup> PhD Candidate, Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran

<sup>a</sup> Professor, Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran

<sup>c</sup> Professor, Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran

Received 17 October, 2013; Revised 20 November, 2013; Accepted 02 October, 2014

## Abstract

Convergent product is an assembly shape concept integrating functions and sub-functions to form a final product. To conceptualize the convergent product problem, a web-based network is considered in which a collection of base functions and sub-functions configure the nodes and each arc in the network is considered to be a link between two nodes. The aim is to find an optimal tree of functionalities in the network adding value to the product in the web environment. First, an algorithm is proposed to assign the links among bases and sub-functions. Then, numerical values as benefits and costs are determined for arcs and nodes, respectively, using a mathematical approach. Also, customer's value corresponding to the benefits is considered. Finally, the Steiner tree methodology is adapted to a multi-objective model optimized by an ant colony optimization method. The approach is applicable for all digital products, such as mobile, tablet, laptop, etc. An example is worked out to illustrate the proposed approach.

**Keywords:** Convergent product, Web-based (digital) network, Multi-objective programming, Steiner tree, Ant colony optimization.

## 1. Introduction

Convergence in electronics and communication sectors has enabled the addition of disparate new functionalities to existing base functions (e.g., adding mobile television to a cell phone or Internet access to a personal digital assistant, PDA). An important managerial issue for such convergent products (CPs) is determination of new functionalities adding more value to a given base. For example, a manufacturer of PDAs may wonder whether it would be a good idea to add satellite radio to it (i.e., a new functionality incongruent with the base), or whether it would be better to add electronic Yellow Pages (i.e., a new functionality congruent with the functions of a PDA). In addition, determining the significance of the base being primarily associated with utilitarian consumption goals (e.g., a PDA), or with hedonic ones (e.g., an MP3 music player) is important.

Convergent product is similar to product assembly where different parts of a product get together to configure a final product. Thus, a designer (modeler) for assembly, as a convergent product, should be able to specify important features affecting the final product. These features may in turn help optimize the manufacturing process.

The paradigms of digital convergence place more emphasis on strategic gravity of convergent products that

are formed by adding new functions to an existing base product (Yoffie, 1997); multiple functions are integrated together in one device to work better rather than they would be delivered separately. Representative examples of this shifting trend are the cases of Apple's iPhone and Microsoft's Xbox. Such convergent products have created new business opportunities for companies to gain or maintain a competitive edge, bringing about immense changes in a wide array of industries (Gill, 2008). Consequently, design of convergent product concepts (CPCs) has likewise become an integral part of business concerns (Gill and Lei, 2009). This is of particular importance in the recent business environments where markets shift rapidly, technologies proliferate unceasingly, thus making business life cycles ever shorter. Systematic design of CPCs needs to address the following analytic issues. The first issue is concerned with the types of data to be employed. The concept design aims to incorporate customer needs into design specs (Callahan and Lasry, 2004). A deeper understanding of the fuzzy front end could help firms to be more successful in their efforts to develop new products (Verworn et al., 2008). Lee et al. (2012) proposed a systematic approach to design of CPCs based on online community information using data mining techniques.

\* Corresponding author Email address: rhz\_1974@ustmb.ac.ir



For instance, the ability of the assembly modeler to furnish information on interferences and clearances between mating parts is particularly useful. Such information would enable the designer to eliminate interference between two mating parts where it is impractical to provide for an interference based on physical assembly requirements. This activity can be accomplished within the modeling program, thereby averting any loss of productivity that might occur due to interferences on the shop floor. Also, knowledge of mass properties for the entire assembly, particularly the center of gravity, may permit the designer to redesign the assembly based on equilibrium and stability considerations. In the absence of such information, the presence of an elevated center of gravity and the attendant instability would only be detected after physical assembly on the shop floor. Three-dimensional exploded views generated by the assembly modeler can help designers verify whether obvious violations of common design for assembly (DFA) guidelines are present, such as absence of chamfers on mating parts.

Corresponding analyses can be achieved within the framework of the assembly modeler. Additionally, the assembly model may be imported into third-party programs that can perform kinematic, dynamic, or tolerance analysis. Tolerance analysis is quite relevant to the physical assembly process. With the input of the assembly model and other user-supplied information such as individual part tolerances, tolerance analysis programs can check the assembly for the presence of tolerance stacks. Tolerance stacks are undesirable elements in the sense that acceptable tolerances on individual parts are combined to produce an unacceptable dimensional relationship, thereby resulting in a malfunctioning or nonfunctioning assembly. Stacks are usually discovered during physical assembly, at which point any remedial procedure becomes expensive in terms of time and cost. Tolerance analysis programs can help the user eliminate or significantly reduce the likelihood of stacks being present.

Based on the results of the tolerance analysis, assembly designs may be optimized by modifying individual part tolerances. Note, however, that tolerance modifications have cost implications; in general, tighter tolerances increase production costs. Engineering handbooks contain tolerance charts indicating the range of tolerances achieved by manufacturing processes such as turning, milling, and grinding. Designers use these tables as guides for rationally assigning part tolerances and selecting manufacturing processes.

A more effective methodology for optimizing product assembly and convergent product is the tree model, whereas the optimization decision is based on a decision tree. One useful tree for assembly modeling as a multiple optimization tool is the Steiner tree.

The Steiner tree problem (STP) is a much actively investigated problem in graph theory and combinatorial optimization. This core problem poses significant

algorithmic challenges and arises in several applications where it serves as a building block for many complex network design problems. Given a connected undirected graph  $G=(V,E)$ , where  $V$  denotes the set of nodes and  $E$  is the set of edges, along with a weight  $C_e$  associated with each edge  $e \in E$ , the Steiner tree problem seeks a minimum-weight subtree of  $G$  that spans a specified subset  $N \subset V$  of *terminal nodes*, optionally using the subset  $N=V-N$  of *Steiner nodes*. The Steiner tree problem is NP-hard for most relevant classes of graphs (Johnson, 1985). Thus, clearly the Steiner tree problem with more objectives is also NP-hard.

The Steiner problem in graphs was originally formulated by Hakimi (1971). Since then, the problem has received considerable attention in the literature. Several exact algorithms and heuristics have been proposed and discussed. Hakimi (1971) remarked that an *Steiner minimal tree* (SMT) for  $X$  in a network  $G=(V,E)$  can be found by enumerating minimum spanning trees of subgraphs of  $G$  induced by supersets of  $X$ . Lawler (1976) suggested a modification of this algorithm, using the fact that the number of Steiner points is bounded by  $|X| - 2$ , showing that not all subsets of  $V$  need to be considered. Restricting NP-hard algorithmic problems regarding arbitrary graphs to a smaller class of graphs will sometimes, yet not always, result in polynomially solvable problems.

Two special cases of the problem,  $N = V$  and  $N = 2$ , can be solved by polynomial time algorithms. When  $N = V$ , the optimal solution of STP is obviously the spanning tree of  $G$  and thus the problem can be solved by polynomial time algorithms such as Prim's algorithm. When  $N = 2$ , the shortest path between two terminal nodes, which can be found by Dijkstra's algorithm, is exactly the Steiner minimum tree.

A survey of Steiner tree problem was given by Hwang and Richards (1992). Several exact algorithms have been proposed such as dynamic programming technique given by Dreyfuss and Wagner (1971), Lagrangean relaxation approach presented by Beasley (1989), branch-and-cut algorithm used by Koch and Martin (1998). Duin and Volgenant (1989) presented some techniques to reduce the size of the graphs for the GSP. Another approach for the GSP is using approximation algorithms to find a near-optimal solution in a reasonable time.

Some heuristic algorithms have been developed such as Shortest Path Heuristic (SPH) given by Takahashi and Matsuyama (1980), Distance Network Heuristic (DNH) presented by Kou et al. (1981), Average Distance Heuristic (ADH) proposed by Rayward-Smith and Clare (1986) and Path-Distance Heuristic (PDH) presented by Winter and MacGregor Smith (1992). Mehlhorn (1988) modified the DNH to make the algorithm faster. Robins and Zelikovsky (2000, 2005) proposed algorithms improving the performance ratio.

Recently, metaheuristics have been considered to arrive at better methods for finding solutions closer to the



optimum. Examples are Genetic Algorithm (GA) (Esbensen, 1995; Kapsalis, et al., 1993), GRASP (Martins et al., 1999) and Tabu search (Ribeiro and Souza, 2000). Although these algorithms have polynomial time complexities, in general, but they cost enormously on large input sets. To deal with the cost issue, some parallel metaheuristic algorithms have been proposed such as parallel GRASP (Martins et al., 1998), parallel GRASP using hybrid local search (Martins et al., 2000) and parallel GA (Fatta et al., 2003).

Some recent researches have been developed in two categories. The first category is related to the theoretical extension of Steiner tree in the field of algorithmic behavior and convergence status (Chakrabarty et al., 2010; Chimani et al., 2012; Gamzu and Segev, 2010; Müller-Hannemann, and Tazari, 2010). The second category is concerned with application of Steiner tree in different aspects of engineering problems (Lu et al., 2011; Zych, and Bilò, 2011).

To produce a new product or to promote an existing one with the idea of using convergent products and development of a mathematical model, keeping base functions and adding sub-functions in satisfying the objectives has not been considered in the literature. In our work here, by applying the Steiner tree, a multi-objective mathematical model is developed to consider promotion of convergent products to satisfy three objectives of cost, benefit and customers' value. The results are some new products with more utility for both the buyer and the producer. Also, to solve the multiobjective Steiner tree model an MOACO is developed.

Here, making use of the Steiner tree, a multi-objective mathematical model is developed for the convergent product. The remainder of our work is organized as follows. In Section 2, the proposed model is described and some useful network algorithms are given. Section 3 presents the mathematical model and an Ant Colony Optimization (ACO) algorithm. Section 4 works out an experimental study to illustrate the proposed algorithm. To evaluate the performance of the proposed MOACO, we develop some medium and large size problems in Section 5. We conclude the study in Section 6.

## 2. The Proposed Model

In our proposed product digital network, a group of functionalities are considered for a product. Customers view their opinions for classifying the functionalities into base functions and sub-functions. We make use of this classification in developing our model. The classification procedure is as follows. First, the customer chooses a product in a list of products being produced in a company. The functionalities of the product are viewed in a web page. Then, the customer clicks either function or sub-function for any of the functionalities. Consequently, customer clicks the "classify" button and observes the classified functionalities in a separate web page. This process is shown in Fig. 1.

Here, we weigh all functionalities (both base functions and sub-functions) considering different significant attributes affecting the value of a product. Therefore, we consider the following mathematical notations.

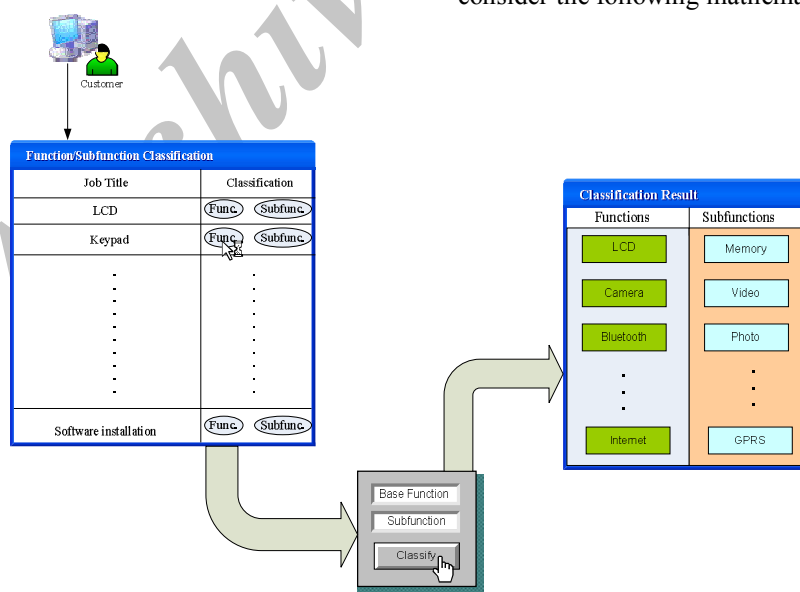


Fig. 1. The classification process



### Mathematical notations:

$i$ and $j$	Index for functions and sub-	and
	functions;	$j=1, \dots, n+m$
$k$	Index for attributes;	$k=1, \dots, p$
$F_{ijk}$	The score of triplet comparison of functions (or sub-functions) with functions (or sub-functions) considering different attributes.	

The three dimension comparison matrix  $F$  is shown in Fig. 2. Note that customers fill in this matrix using numerical values  $F_{ijk} \in [0,1]$ .

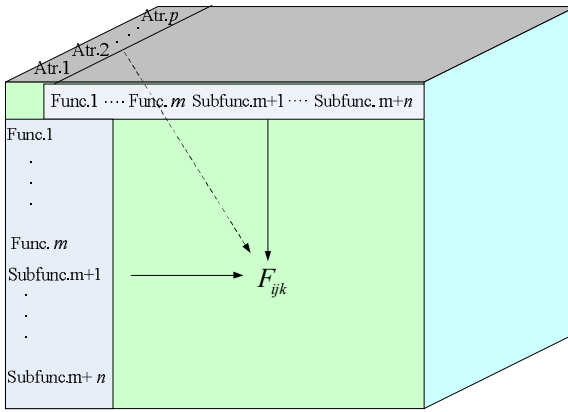


Fig. 2. The three dimensional comparison matrix

This matrix is normalized to remove the scales. The normalized values are shown by  $F_{ijk}^{norm}$ . A threshold value

of  $\theta$  is considered in a way that the  $f_{ij} = \frac{\sum_{k=1}^p F_{ijk}^{norm}}{p} \geq \theta$

are chosen to be assigned as links. The  $f_{ij}$  is a value that customer considers for arc  $(i, j)$ . These links configure a network called purified network as shown in Fig. 3.

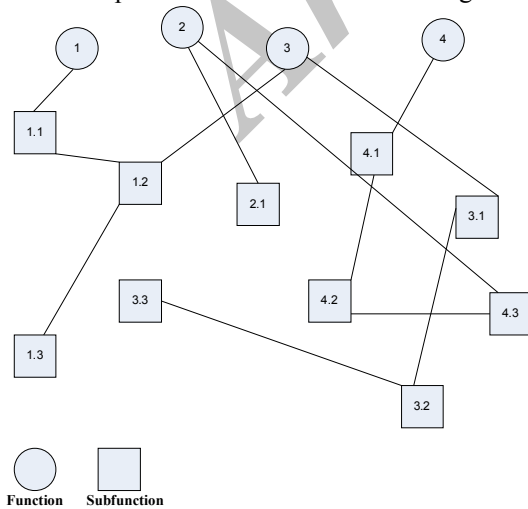


Fig. 3. A purified network

Now, using the purified network, we characterize the arcs. To do this, two processes of leveling and clustering are performed. For leveling, we set the base functions at level zero, sub-functions with one outlet to the previous level at level 1, and so on. Thus, an  $l$  level network is configured. The proposed algorithm is given next.

**Algorithm 1:** Leveling to configure a leveled network.

**Step 0:** Set the base functions at level 0. Let  $l=0$ .

**Step 1:** **While** sub-functions exist for processing **do**

Find sub-functions with a link to a function (or sub-function) at level  $l$  and put them in level  $l+1$ . Let  $l=l+1$ ;

**End while.**

{ $l$  is the number of levels.}

**Step 2:** **Stop.**

The nodes of leveled network are associated with given costs. We are looking for the benefit each link provides. Here, a clustering approach is considered. Clusters are formed as follows: at each level, all sub-functions linked to a single parent is grouped in a cluster. Therefore, clusters consisting different nodes are configured. These clusters are being configured as a new network. The leveling and clustering processes are shown schematically in Fig. 4. Later, we apply the Steiner tree methodology to optimize this network.

The proposed algorithm for clustering is given next.

**Algorithm 2:** Clustering of levels in a leveled network.

**Step 0:** Set each node at level 0 to be a cluster.

**Step 1:** **For**  $i=1$  till  $l$  **do**

{Form clusters at level  $i$ }

Cluster all sub-functions at level  $i$  linked to a single parent at level  $i-1$ ;

Solve a zero/one mathematical program for level  $i$  (we will discuss the corresponding mathematical program later on);

Perform purification of benefits and costs at level  $i$  (as discussed later on);

**End for.**

**Step 2:** **Stop.**

Here, the clustered network is used to configure a tree (the Steiner tree) keeping the base functions and optimizing three objectives of minimal cost, maximal profit and the maximum of total values that customers consider for existing arcs in the convergent product value adding process. In traditional Steiner tree approach, the aim is usually to find a tree having a minimal arc total cost. Here, we extend the approach by looking for a tree having the base functions and meanwhile minimize cost, maximize benefit and maximize customer's total value. In fact, the model structure's closeness to the Steiner tree model justifies modeling the problem with the proposed approach. Next, we formulate our adapted proposed Steiner tree model. In the proposed network, node  $i$  (function or sub-function  $i$ ) have two costs:



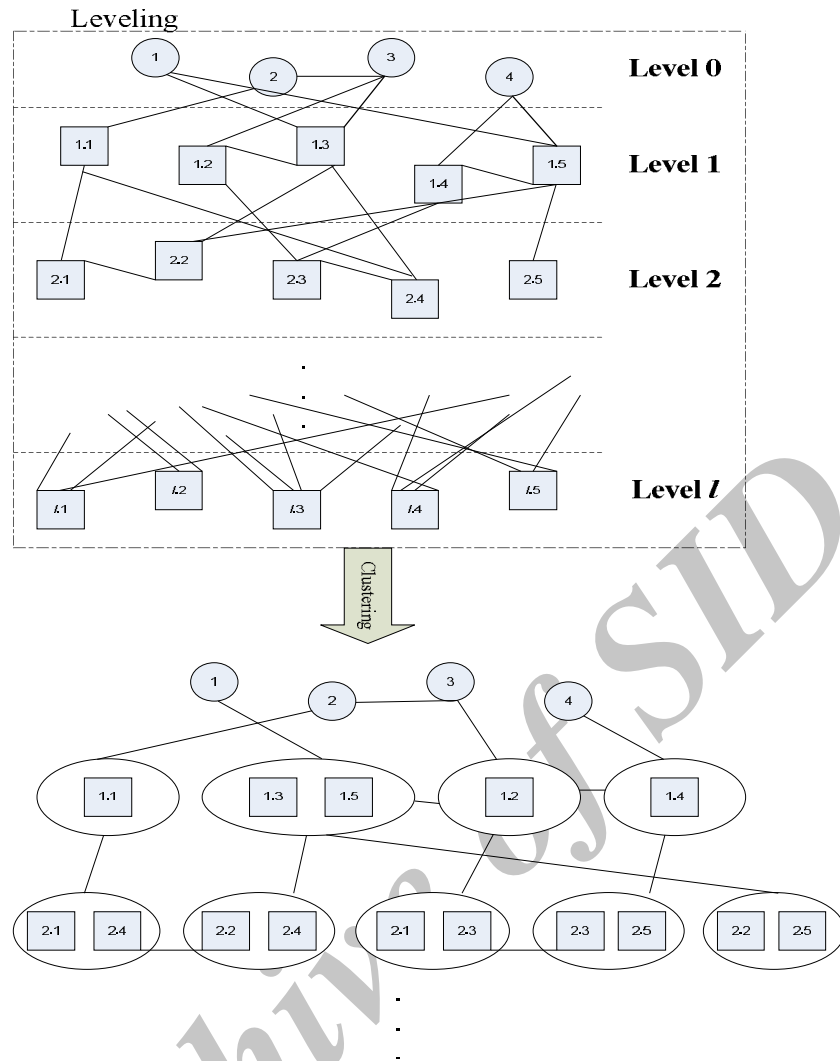


Fig. 4. Leveling and clustering processes

$c_{i1}$ : software cost,  
 $c_{i2}$ : hardware cost.

Each arc is accompanied with a benefit  $p_{ii'}$ , attained by nodes  $i$  and  $i'$ . Regarding the solution approach and using the Steiner tree in the proposed network and the NP-hardness of the problem, we used leveling and clustering processes to reduce the complexity of the problem. In clustering, it is not acceptable for any node to be included in more than one cluster at any level. To guarantee this, for each level  $l$ , a zero/one mathematical program is developed in order to properly appropriate nodes to clusters with the aim of minimizing the total cost.

Next, we give the zero/one mathematical program and the purification procedure for each level.

**The zero/one mathematical program for level  $l$ :**

$$\min \quad T = \sum_{i \in n_l} \sum_{j \in m_{li}} (\alpha_{ij} c_{ij1} + \beta_{ij} c_{ij2}) z_{ij} \quad (1)$$

$$\sum_{j \in m_{li}} z_{ij} = 1, \quad i = 1, \dots, n_l$$

$$z_{ij} = \begin{cases} 1, & \text{if node } i \text{ is in cluster } j \\ 0, & \text{otherwise,} \end{cases}$$

where,  $\beta_{ij}, \alpha_{ij} \in [0, 1], \forall i, j$ , and

$m_{li}$ : set of indices of clusters at level  $l$  where node  $i$  is included,

$n_l$ : set of indices of different nodes at level  $l$ ,

$c_{ij2}$ : hardware cost of node  $i$  in cluster  $j$  ( $c_{ij2} = c_{i2}$ , for all  $i$  and  $j$ ),

$c_{ij1}$ : software cost of node  $i$  in cluster  $j$  ( $c_{ij1} = c_{i1}$ , for all  $i$  and  $j$ ),

$\alpha_{ij}$ : the software reduction cost coefficient of node  $i$  in cluster  $j$ ,

$\beta_{ij}$ : the hardware reduction cost coefficient of node  $i$  in cluster  $j$ .

**Purifying benefits, costs and customers total value at level  $l$ :**

To determine the cost for each cluster at level  $l$ , we use



$$c_{jl} = \sum_{i \in n_l} (\alpha_{ij} c_{ij1} + \beta_{ij} c_{ij2}) - \sum_{\forall i, i'} p_{ii'}, \quad (2)$$

with  $c_{ij1}, c_{ij2}$  and  $n_l$  as defined above,

$\beta_{ij}, \alpha_{ij} \in [0, 1], \forall i, j$ , and

$c_{jl}$ : cost of cluster  $j$  at level  $l$ ,

$\alpha_{ij}$ : the software reduction cost coefficient of node  $i$  in cluster  $j$ ,

$\beta_{ij}$ : the hardware reduction cost coefficient of node  $i$  in cluster  $j$ ,

$p_{ii'}$ : the benefit of an arc connecting node  $i$  in cluster  $j$  to node  $i'$  in cluster  $j'$ .

Also, to adjust the combined arc benefits in clusters, the following equation is used:

$$p_{jj'} = (1 + \gamma_{jj'}) \sum_{\forall i, i'} p_{ii'}, \quad (3)$$

where,  $\forall i, i', j, j'$ , and

$p_{jj'}$ : the adjusted arc benefit connecting cluster  $j$  to cluster  $j'$ ,

$p_{ii'}$ : the benefit of an arc connecting node  $i$  in cluster  $j$  to node  $i'$  in cluster  $j'$ ,

$\gamma_{jj'}$ : the added value configured from nodes in clusters  $j$  and  $j'$ .

Also, to adjust the combined arc customers total value in clusters, the following equation is used:

$$f_{jj'} = (1 + \gamma_{jj'}) \sum_{\forall i, i'} f_{ii'}, \quad (4)$$

where,  $\forall i, i', j, j'$ , and

$f_{jj'}$ : the adjusted arc customers total value connecting cluster  $j$  to cluster  $j'$ ,

$f_{ii'}$ : the customers total value of an arc connecting node  $i$  in cluster  $j$  to node  $i'$  in cluster  $j'$ ,

$\gamma_{jj'}$ : the added value configured from nodes in clusters  $j$  and  $j'$ .

Algorithms 1 and 2 are transformed into Algorithm 3 using the aforementioned considerations. Also, each node should be in only one cluster at level  $l$ . The node having a minimal cost is chosen for the level  $l$ . Then, instead of using the zero/one mathematical program for level  $l$ , we can use step 3 of Algorithm 3. This leads a reduction of computations by avoiding the need for using the zero/one programs.

**Algorithm 3:** leveling and clustering in the network.

**Step 0:** Set the base functions at level 0. Let  $l=0$ .

**Step 1: While** sub-functions exist for processing **do**

Find sub-functions with a link to a function (or sub-function) at level  $l$  and place them at level  $l+1$ ; Let  $l=l+1$ ;

**End while.**

$\{l$  is the number of levels $\}$

**Step 2:** Set each node at level 0 to be a cluster.

**Step 3: For**  $i=1$  till  $l$  **do**

$\{$ Form clusters at level  $i$  $\}$

Cluster all sub-functions at level  $i$  linked to a single parent at level  $i-1$ ;

**While**  $|n_i| > 0$  **do**

Select  $k \in n_i$  such that

$$\alpha_{kp} c_{kp1} + \beta_{kp} c_{kp2} = \min_{j \in m_{ik}} \{ \alpha_{kj} c_{kj1} + \beta_{kj} c_{kj2} \}.$$

Set  $z_{kp} = 1$

and  $z_{kj} = 0, \forall j \in m_{ik}, j \neq p$ ;

$n_i \leftarrow n_i - \{k\}$ .

**End while;**

**For**  $j=1$  till  $q_i$  **do**  $\{q_i$  is the number of clusters in the level  $i\}$

$$c_{ji} = \sum_{i \in n_l} (\alpha_{ij} c_{ij1} + \beta_{ij} c_{ij2}) - \sum_{\forall i, i'} p_{ii'};$$

**End for;**

**For**  $j=1$  till  $q_i$  **do**

**For**  $j'=1$  till  $q_i$  **do**

$$p_{jj'} = (1 + \gamma_{jj'}) \sum_{\forall i, i'} p_{ii'};$$

$$f_{jj'} = (1 + \gamma_{jj'}) \sum_{\forall i, i'} f_{ii'};$$

**End for;**

**End for;**

**End for.**

**Step 4: Stop.**

### 3. Mathematical Formulation and the Extended MOACO Approach

Here, we first propose the mathematical model for the considered problem and then state the solution approach.

#### 3.1. Mathematical formulation

We first recall the undirected Dantzig–Fulkerson–Johnson model for the convergent product Steiner tree problem (CPSTP) presented in (Costa et al., 2006). Let  $x_{ij}$  and  $y_i$  be binary variables associated with links  $(i, j) \in E$  and clusters  $i \in V$ , respectively. Variable  $y_i$  is 1 if cluster  $i$  belongs to the solution, and is 0 otherwise. Similarly, variable  $x_{ij}$  is 1 if link  $(i, j)$  belongs to the solution, and is 0 otherwise. For  $S \subseteq V$ , define  $E(S)$  as the set of links with both end nodes in  $S$ . Assume that terminals are the set  $N$ . The mathematical model can then be written as:



$$\text{Maximize } \sum_{(i,j) \in E} p_{ij} \cdot x_{ij}, \quad (5)$$

$$\text{Maximize } \sum_{(i,j) \in E} f_{ij} \cdot x_{ij}, \quad (6)$$

$$\text{Minimize } \sum_{i \in V} c_i \cdot y_i, \quad (7)$$

Such that

$$\sum_{(i,j) \in E} x_{ij} = \sum_{i \in V} y_i - 1, \quad (8)$$

$$\sum_{(i,j) \in E(S)} x_{ij} \leq \sum_{i \in S - \{k\}} y_i$$

$$, \quad \forall k \in S \subseteq V, \quad \forall S : |S| \geq 2, \quad (9)$$

$$y_h = 1, \quad \forall h \in N, \quad (10)$$

$$x_{ij} \in \{0,1\}, \quad \forall i, j \in E, \quad (11)$$

$$y_i \in \{0,1\}, \quad \forall i \in V. \quad (12)$$

The objectives are to maximize the aggregated benefits, minimize the aggregated costs and maximize the aggregated customers total value. Constraint (8) guarantees that the number of clusters in a solution is equal to the number of links minus one, and constraints (9) are the connectivity constraints. The number of

constraints in (9) equals  $2^{|V|} - |V| - 1$ . As a result, the number of variables and constraints are increased exponentially with respect to the number of clusters. Constraints (10) impose the terminal clusters to exist in the tree. Relations (11) and (12) show the variable types.

Assume that two products  $A$  and  $B$  have respective costs  $c_A$  and  $c_B$ , respective benefits  $p_A$  and  $p_B$ , and respective customers' total value  $v_A$  and  $v_B$ . If

$c_A < c_B$ ,  $p_A > p_B$  and  $v_A > v_B$ , then product  $A$  dominates product  $B$ , but if  $c_A < c_B$ ,  $p_A < p_B$  and  $v_A < v_B$  or

$c_A < c_B$ ,  $p_A < p_B$  and  $v_A > v_B$  or ..., then neither  $A$  dominates  $B$  nor  $B$  dominates  $A$ . Since neither is a dominant solution,  $A$  and  $B$  are Pareto solutions for the problem.

The two objectives, benefit and customers' total value, are to maximize and it is possible to aggregate them as a single objective. But, aggregating these two objectives leads to underestimation of customers' total value due to differences in the coefficients. This is a cause for missing some Pareto optimal solutions.

### 3.2. The extended MOACO approach

Ant colony optimization (ACO) was first introduced by Dorigo et al. (1991) for solving the traveling salesman

problem. This method of optimization is inspired by the behavior of various ants as they are searching for the shortest path among the possible paths in finding the food sources. To solve a problem with an ACO algorithm, the problem is usually defined and represented by a graph.

As ants start the production of solutions in the graph, they are guided by pheromone trails and heuristic information. The heuristic information is the measure of preference to move from state  $S_i$  to  $S_j$  (in the graph, from node  $i$  to node  $j$ ) and the pheromone trails are the amount of pheromone deposited by ants in the previous stages showing the learning desirability of moving from state  $S_i$  to  $S_j$ . Considering the solution found. While ants update the amount of pheromone trails, the nodes with more amounts of pheromone trails are more likely to be selected by the artificial ants.

In recent years, many researchers have developed various ACO algorithms for combinatorial problems such as vehicle routing, traveling sales man, production scheduling, sequential ordering, telecommunication routing, etc. Various ACO algorithms are different in the procedures for updating pheromone trails, evaporation and transition rules. Ant System (AS) was first developed by Dorigo et al. (1991) and applied to solve the classical traveling salesman problem. Six years later, Dorigo and Gambardella (1997) introduced another algorithm which performed better than AS and was called ant colony system (ACS). ACS uses different procedures in local and global pdating of the pheromone trails as well as the transition rules. There are different versions of ACO algorithms such as Elitist Ant System (Stützle and Dorigo, 1999), rank-based AS (Stützle and Dorigo (1999; Dorigo et al., 1999) and Min-Max AS (Stützle and Dorigo, 1999; Socha et al., 2003). Ant colony algorithms have successfully been used to solve single objective combinatorial problems (see, for example, Dorigo and Gambardella, 1997; Colorni and Dorigo, 1994; Gambardella and Dorigo, 2000). In recent years, to elevate the performance level, studies on hybrid ACO algorithm with other meta-heuristics such as genetic and other evolutionary algorithms (Jangam and Chakraborti, 2007) have been made.

Regarding the success of ACO in the single objective case, researchers have been interested to employ ACO for solving multi-objective problems. They designed various kinds of algorithms by modification of both AS and ACS. Because of the excellent speed of ACO in constructing feasible solutions, it was employed as an alternative tool in place of an exact method for solving multi-objective problems. ACO appears to be proper for solving multi-objective problems due to its high speed in generating solutions leading to detection of more non-dominated solutions. Iredi et al. (2001) developed a multi-objective ACO algorithm on the basis of AS for bi-criteria vehicle routing problem. They used two pheromone trails, one for each objective. They also combined the information of pheromone trails to calculate the probability distribution of the transition rule. In order to force the ants to search in



different regions of the solution space, they defined a specific parameter and used it as a weighting factor in the transition rule. Doerner et al. (2004) designed a multi-objective ACO algorithm on the basis of ACS for multi-objective portfolio selection problem. They considered several pheromone matrices, one for each objective. The ants used the maximum selection method in selecting the next node combining the pheromone trails information of different objectives to calculate the probability distribution of the transition rule. At the end of each iterations, best ants and second-best ants generated in the current iteration updated the pheromone proposed trails. Multiple ant colony system proposed by Baran and Schaerer (2003) was devised for vehicle routing problem with time windows based on ACS algorithm using only one pheromone trail matrix and several heuristic parameters, one parameter corresponding to each objective. The transition rule by Baran and Schaerer (2003) uses the information of pheromone trail matrix and some heuristic parameters. Also, having parameters specific to the indices of the ants, accommodates for the ants to search in different regions of the solution space. Doerner et al. (2003) proposed the COMPET ants algorithm for bi-objective transportation problem. Their formula works on the basis of rank-based AS and uses two colonies of ants, two pheromone trail matrices and two heuristic parameters, one for each objective. The distinction in this algorithm is that the number of ants in each colony is determined using the solutions constructed by the ants of the colonies. Colonies constructing better solutions get more ants for the next iteration. In this algorithm, some ants in colonies called spy ants mix the pheromone trails and heuristic information of all the colonies to search in central areas of the Pareto frontier. Garcia-Martinez et al. (2007) recently presented a taxonomy and review of existing multi-objective ant colony optimization algorithms as well as an empirical analysis of performance of these algorithms in comparison with multi-objective genetic algorithms. Considering the literature reviewed, the multi-objective ACO algorithms can be classified in 3 classes:

- Algorithms that employ several colonies, one for each objective.
- Algorithms that employ several pheromone trail matrices, one for each objective.
- Algorithms that employ several heuristic information, one for each objective.

Here, our proposed ACO algorithm for solving MOSTP problems makes use of the second approach being and adaptive modified ant colony system algorithm to solve the multi-objective problem.

#### Ant colony functions

The suggested ant colony algorithm is composed of some main elements such as pheromone matrix, initializations and heuristic parameters, transition rule, storage of non-dominated solutions, local and global updates. We described these elements in this section. The algorithm is designed based on the classical ACS proposed by Dorigo

and Gambardella (1997) with maximum selection in the transition rule as well as local and global updates.

**Pheromone matrix:** In the algorithm, three pheromone matrices are defined, one for each objective. At the end of each iteration, the pheromone matrices are updated separately considering the solutions generated up to the current iteration. The ideas of using two or more pheromone trail matrices is have been promoted by some researchers such as Iredi et al. (2001) for the bi-criteria vehicle routing problem, Cardoso et al. (2003) for the network traffic optimization and Doerner et al. (2003) for the bi-objective transportation problem.

**Initialization and heuristic parameter:** In the Proposed Initialization and heuristic parameter are used from the approach developed by Gosavi (2003). The extended algorithm consists of several trials, where each trial results in a generated candidate Steiner tree. The situation is much like the original ant colony systems approach, where in each iteration an ant traces out a cyclic tour for the traveling salesperson problem (Dorigo and Maniezzo, 1996; Dorigo and Gambardella, 1997). However, in order to generate a Steiner tree, we have to make use of multiple ants. This is because a Steiner tree, unlike a single cyclic path solution that the traveling sales man problem warrants, consists of a tree having more than a single path. We reason that a good way to generate a tree in a given graph is to make use of several ants that cooperatively generate a tree, whose separate branches are defined by the paths traveled by each ant.

One ant is placed initially at every one of the given terminal vertices that are to be connected. In each iteration, an ant is moved to a new location via an edge. The new location is determined stochastically, but is biased in such a manner that the ants get drawn to the paths traced out by one another. Each ant maintains its own separate list of vertices already visited. In conforming to the parlance used in the original work (Dorigo and Maniezzo, 1996), we call this list the ant's *tabu list*. This list is maintained to prevent it from entering such a vertex again, which would otherwise produce a cycle. When any ant collides with another ant, or even with the path of another, it merges into the latter. This is because paths followed by the two joining ants become connected into one single sub-tree. This complex interaction between the ants amounts to the Steiner tree being defined in each trial by their paths, when all ants merge into a single entity.

An ant  $m$ , currently at a vertex  $I$ , selects a vertex  $j$  not in its tabu list  $T^{(m)}$ , to move to, only if  $(i, j) \in E$ . We define two potentials for each vertex  $j$  in  $V$  with respect to an ant  $m$  as follows:

$$\psi_j^{(m)} = \max_k (P_{jk}), \quad (13)$$

and

$$\delta_j^{(m)} = \max_k (F_{jk}), \quad (14)$$

where,



$$k \in \bigcup_{m' \neq m} T^{(m')}.$$

Here,  $m'$  is any other ant,  $P_{jk}$  (sum of all benefits) and  $F_{jk}$  (sum of all customers values) are the longest distance from the two vertices  $j$  and  $k$ . These distances are computed as the sum of the weights of all the edges leading from  $j$  to  $k$ . The longest distances between the vertices can easily be computed using the well-known Floyd all-pair shortest path algorithm, a dynamic programming approach.

We define the desirability of a vertex with respect to an ant  $m$ , currently in vertex  $I$ , as follows:

$$\eta_{ij}^{(m)} = \frac{p_{ij} + \gamma \psi_j^{(m)}}{H}, \quad (15)$$

$$\theta_{ij}^{(m)} = \frac{f_{ij} + \kappa \delta_j^{(m)}}{K} \quad (16)$$

$$\varphi_j^{(m)} = \frac{1}{c(j)}, \quad (17)$$

where, the quantities  $\gamma$  and  $\kappa$  are constant

and  $H = \sum_{i=1}^n \sum_{j=1}^n p_{ij}$ ,  $K = \sum_{i=1}^n \sum_{j=1}^n f_{ij}$ . The algorithm

biases the ants towards choosing edges with more desirability.

**Local update and global update:** The trail updating are performed at the end of the trails. The trail updating follows the simple rules,

$$\begin{aligned} \tau_{ij} &= (1 - \rho) \tau_{ij} + \rho \Delta \tau_{ij} \\ \tau'_{ij} &= (1 - \rho) \tau'_{ij} + \rho \Delta \tau'_{ij} \end{aligned} \quad (18)$$

$$\mu_j = (1 - \rho) \mu_j + \rho \Delta \mu_j,$$

where  $\rho$  is a parameter between 0 and 1, called the trail evaporation rate, and measures how rapidly the trail evolves. The local update (evaporation) procedure prevents the algorithm to rapidly converge to a local optimal solution and forces the artificial ants to explore new areas of the search space by efficiently disregarding the previous information. The general form of the incremental updates  $\Delta \tau_{ij}$ ,  $\Delta \tau'_{ij}$  and  $\Delta \mu_j$  are given by

$$\Delta \tau_{ij} = \begin{cases} Q.w(P), & \text{if } (i,j) \in E_S \\ 0, & \text{otherwise} \end{cases} \quad (19)$$

where  $w(P) = \frac{\sum p_{ij}}{H}$  such that  $(i,j) \in E_S$ ,

$$\Delta \tau'_{ij} = \begin{cases} Q.w(F), & \text{if } (i,j) \in E_S \\ 0, & \text{otherwise} \end{cases} \quad (20)$$

where  $w(F) = \frac{\sum f_{ij}}{K}$  such that  $(i,j) \in E_S$ , and

$$\Delta \mu_j = \begin{cases} \frac{Q}{w(C)}, & \text{if } j \in V_S \\ 0, & \text{otherwise,} \end{cases} \quad (21)$$

where  $w(C) = \sum c_j$  such that  $j \in V_S$ .

We have carried out two forms of updating given in Dorigo and Gambardella (1997) as follows. In the *local updating* rule, the Steiner tree  $S$ , which is used to determine the incremental updates  $\Delta \tau_{ij}$ ,  $\Delta \tau'_{ij}$

and  $\Delta \mu_j$ , is the one computed in the current trial. In other words,  $S$ , in equations (19), (20) and (21) are set to  $S_{current}$ . We maintain the quantity  $Q$  at a constant level. But, we also use a *global updating* of the trails also, where the incrementing operation is carried out only for those edges and nodes that are part of the non-dominate solutions obtained from the beginning of the iteration until the last trail.

**Transition rule:** The following probability distribution shows how an ant moves from node  $i$  to its subsequent node:

**If**  $q \leq q_0$  **then**

$$\text{let } p_{ij} = \begin{cases} 1, & \text{if } j = \arg \max_{u \in w(i)} \left\{ \begin{aligned} &(\tau_{ij})^\alpha (\eta_{ij}^m)^\beta \cdot (\tau'_{ij})^\alpha (\theta_{ij}^m)^\beta \cdot (\mu_j)^\alpha (\phi_j^m)^\beta \end{aligned} \right\} \\ 0, & \text{otherwise,} \end{cases} \quad (22)$$

**Else**

**let**

$$p_{ij} = \begin{cases} \frac{(\tau_{ij})^\alpha (\eta_{ij}^m)^\beta \cdot (\tau'_{ij})^\alpha (\theta_{ij}^m)^\beta \cdot (\mu_j)^\alpha (\phi_j^m)^\beta}{\sum_{u \in w(i)} (\tau_{iu})^\alpha (\eta_{iu}^m)^\beta \cdot (\tau'_{iu})^\alpha (\theta_{iu}^m)^\beta \cdot (\mu_u)^\alpha (\phi_u^m)^\beta}, & \text{if } j \in w(i) \\ 0, & \text{otherwise,} \end{cases} \quad (23)$$

where  $w(i) = \bigcup_{\forall m} T^{(m)}$  is a set of outgoing nodes from node  $i$  not previously visited by the ants.

**Keeping non-dominated solutions procedure:** In each iteration of the algorithm, each ant generates a solution. If the new generated solution is a non-dominated solution, the Pareto optimal set should be updated. Fig. 5 presents a flowchart for the steps of the algorithm.



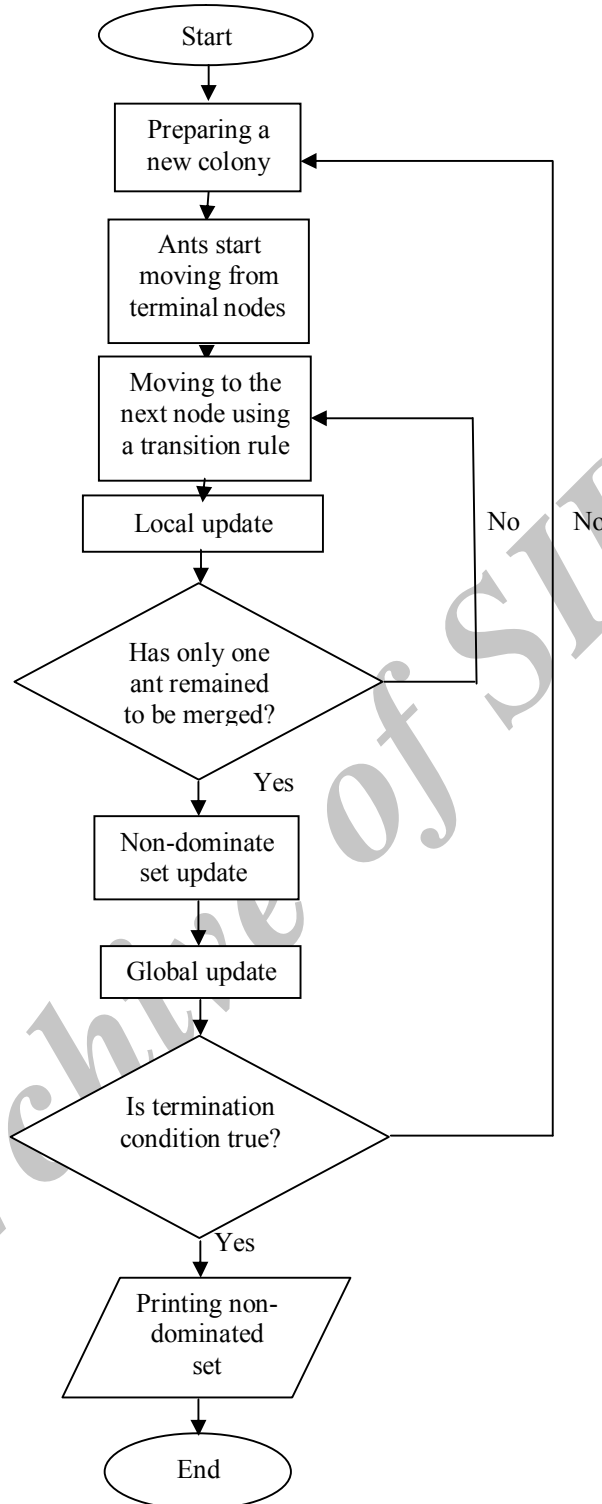


Fig. 5. Flow chart of the multi-objective ant colony algorithm

#### 4. Experimental Study

Here, to illustrate the applicability and effectiveness of our proposed multiple optimization process, an experiment is worked out. Consider an undirected graph  $G=(i, j)$  with the cluster set  $V = \{1, \dots, n\}$  and the link

set  $E = \{e = (i, j) : i, j \in V, i < j\}$ , non-negative profits,  $p_e$ , associated with the links and non-negative costs,  $c_i$ , associated with the clusters. In this Steiner tree problem, the aim is to find the tree maximizing the revenue, i.e., the sum of the profits of the links in  $p_e$  spanned by the solution, and minimizing the sum of the costs of the clusters in the solution. On the one hand, we



would like to have solution spanning all links avoiding the loss of profit; but this can be too expensive in terms of the cost of the tree-structured network providing service to all clusters. Thus, there is a trade-off between the cost of the clusters being in the solution and the profit of the links by the solution.

The three dimensional matrix of functions, sub-functions, and attributes are shown in Table 1. Note that the tables related to all the three attributes are configured and their arithmetic means are shown as the final functions, sub-functions, and attribute comparison matrix.

Our threshold value is considered to be 0.561 which is the mean of the data given in Table 1. Therefore, the thresholded matrix is shown in Table 2, and the corresponding network is configured as Fig. 6.

Table 1  
The three dimensional comparison matrix for all the attributes

Attributes	B <sub>1</sub>	B <sub>2</sub>	B <sub>3</sub>	S <sub>1</sub>	S <sub>2</sub>	S <sub>3</sub>	S <sub>4</sub>	S <sub>5</sub>	S <sub>6</sub>	S <sub>7</sub>
B <sub>1</sub>	0	0.59	0.58	0.6	0.53	0.66	0.46	0.52	0.48	0.46
B <sub>2</sub>	-	0	0.46	0.36	0.46	0.63	0.86	0.58	0.43	0.43
B <sub>3</sub>	-	-	0	0.59	0.56	0.53	0.9	0.63	0.33	0.53
S <sub>1</sub>	-	-	-	0	0.58	0.59	0.6	0.53	0.6	0.6
S <sub>2</sub>	-	-	-	-	0	0.36	0.43	0.43	0.7	0.83
S <sub>3</sub>	-	-	-	-	-	0	0.63	0.6	0.63	0.58
S <sub>4</sub>	-	-	-	-	-	-	0	0.46	0.6	0.43
S <sub>5</sub>	-	-	-	-	-	-	-	0	0.65	0.63
S <sub>6</sub>	-	-	-	-	-	-	-	-	0	0.63
S <sub>7</sub>	-	-	-	-	-	-	-	-	-	0

Table 2  
The thresholded comparison matrix for all the attributes

Attributes	B <sub>1</sub>	B <sub>2</sub>	B <sub>3</sub>	S <sub>1</sub>	S <sub>2</sub>	S <sub>3</sub>	S <sub>4</sub>	S <sub>5</sub>	S <sub>6</sub>	S <sub>7</sub>
B <sub>1</sub>	0	1	1	1	0	1	0	0	0	0
B <sub>2</sub>	-	0	0	0	0	1	1	1	0	0
B <sub>3</sub>	-	-	0	1	0	0	1	1	0	0
S <sub>1</sub>	-	-	-	0	1	1	1	0	1	1
S <sub>2</sub>	-	-	-	-	0	0	0	0	1	1
S <sub>3</sub>	-	-	-	-	-	0	1	1	1	1
S <sub>4</sub>	-	-	-	-	-	-	0	0	1	0
S <sub>5</sub>	-	-	-	-	-	-	-	0	1	1
S <sub>6</sub>	-	-	-	-	-	-	-	-	0	1
S <sub>7</sub>	-	-	-	-	-	-	-	-	-	0

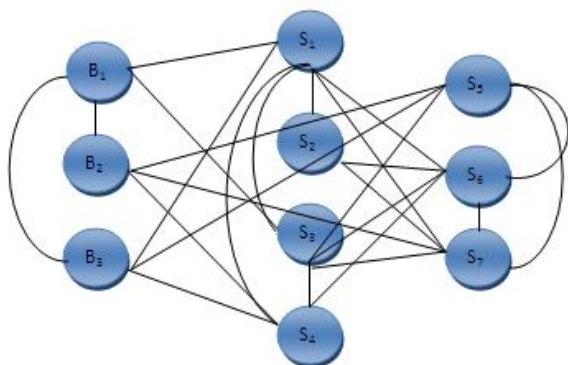


Fig. 6. The configured thresholded network

Then, the leveling process (the zero th and the first steps of Algorithm 3) is performed and the leveled network is configured as Fig. 7. The clustered network (the second and the third steps of Algorithm 3) is shown in Fig. 8.

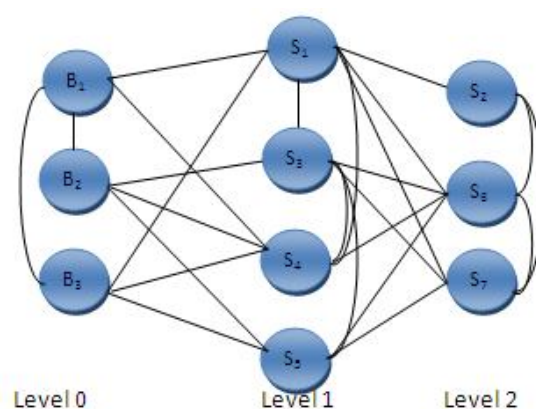


Fig. 7. The configured leveled network

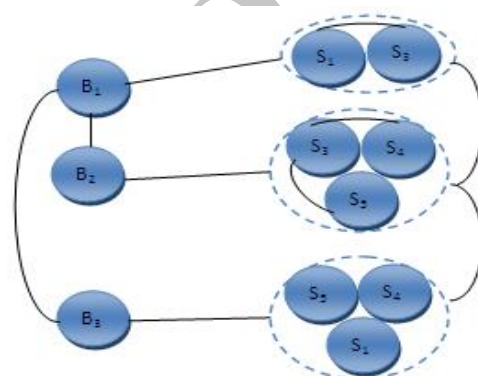


Fig. 8. The configured clustered network (the second and the third steps of Algorithm 3)

The cost vectors, the benefit matrix, customers total value matrix and the matrices  $\alpha = [\alpha_{ij}]$ ,  $\beta = [\beta_{ij}]$  are obtained to be:

$C_1 = (150, 210, 180, 20, 30, 30, 50, 10, 20, 20)$ ,  
 $C_2 = (300, 450, 600, 70, 80, 50, 40, 20, 20, 20)$ .

$$\alpha = \begin{bmatrix} - & - & - \\ - & - & - \\ - & - & - \\ 0.65 & - & 0.7 \\ - & - & - \\ 1 & 0.8 & - \\ - & 0.6 & 0.9 \\ - & 0.7 & 0.65 \\ - & - & - \\ - & - & - \end{bmatrix} \quad \beta = \begin{bmatrix} - & - & - \\ - & - & - \\ - & - & - \\ 0.9 & - & 0.9 \\ - & - & - \\ 0.9 & 0.7 & - \\ - & 0.6 & 0.9 \\ - & 0.7 & 0.6 \\ - & - & - \\ - & - & - \end{bmatrix}$$



$$p = \begin{bmatrix} - & 1500 & 1300 & 100 & - & 80 & - & - & - & - \\ - & - & - & - & - & 70 & 90 & 120 & - & - \\ - & - & - & 60 & - & - & 100 & 110 & - & - \\ - & - & - & - & 50 & 90 & 110 & - & 60 & 40 \\ - & - & - & - & - & - & - & - & 70 & 30 \\ - & - & - & - & - & 80 & 70 & 40 & 20 & - \\ - & - & - & - & - & - & - & - & 30 & - \\ - & - & - & - & - & - & - & - & 20 & 10 \\ - & - & - & - & - & - & - & - & - & 20 \\ - & - & - & - & - & - & - & - & - & - \end{bmatrix}$$

$$f = \begin{bmatrix} - & 0.59 & 0.58 & 0.6 & - & 0.66 & - & - & - & - \\ - & - & - & - & - & 0.63 & 0.86 & 0.58 & - & - \\ - & - & - & 0.59 & - & - & 0.9 & 0.63 & - & - \\ - & - & - & - & 0.58 & 0.59 & 0.6 & - & 0.6 & 0.6 \\ - & - & - & - & - & - & - & - & 0.7 & 0.83 \\ - & - & - & - & - & 0.63 & 0.6 & 0.63 & 0.58 & - \\ - & - & - & - & - & - & - & 0.6 & - & - \\ - & - & - & - & - & - & - & 0.65 & 0.63 & - \\ - & - & - & - & - & - & - & - & 0.63 & - \\ - & - & - & - & - & - & - & - & - & - \end{bmatrix}$$

For level 1, using iteration 1 of the while loop in step 3 of Algorithm 3, we obtain:

$$(\alpha_{41}c_{411} + \beta_{41}c_{412}) = 76$$

$$(\alpha_{43}c_{431} + \beta_{43}c_{432}) = 77$$

$$(\alpha_{61}c_{611} + \beta_{61}c_{612}) = 75$$

$$(\alpha_{62}c_{621} + \beta_{62}c_{622}) = 59$$

$$(\alpha_{72}c_{721} + \beta_{72}c_{722}) = 54$$

$$(\alpha_{73}c_{731} + \beta_{73}c_{732}) = 81$$

$$(\alpha_{82}c_{821} + \beta_{82}c_{822}) = 21$$

$$(\alpha_{83}c_{831} + \beta_{83}c_{832}) = 18.5$$

Therefore,  $z_{41} = 1, z_{62} = 1, z_{72} = 1$  and  $z_{83} = 1$  with other variables equal to zero. The configured network up to level 1 is shown in Fig. 9.

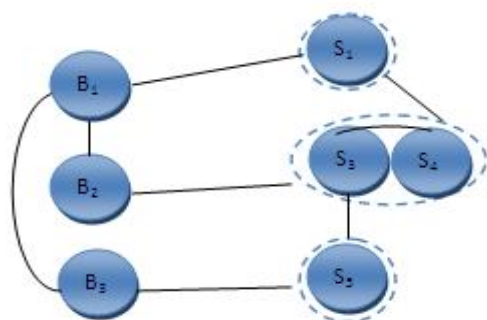


Fig. 9. The configured clustered network for level 1

The next iteration of Algorithm 3 for clustering is performed, and the purified network is obtained as Fig. 10.

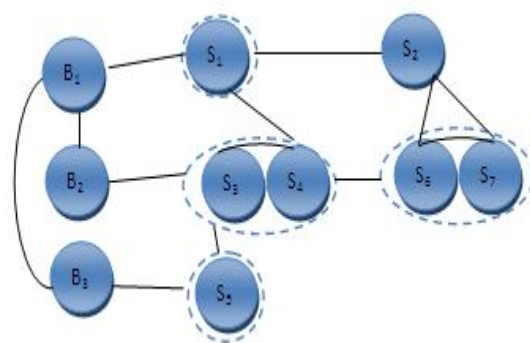


Fig. 10. The configured clustered network

After purifying the benefits, costs and customers total value for level 2, the final cost, benefit and customers total value matrices are formed as follows:

$$\tilde{p} = \begin{bmatrix} - & 1500 & 1300 & 100 & - & - & - & - \\ - & - & - & - & - & 208 & - & - \\ - & - & - & - & - & - & 110 & - \\ - & - & - & 50 & 132 & - & - & - \\ - & - & - & - & - & - & - & 130 \\ - & - & - & - & - & - & 84 & 135 \\ - & - & - & - & - & - & - & - \\ - & - & - & - & - & - & - & - \end{bmatrix}$$

$$\tilde{c} = (450 \quad 660 \quad 780 \quad 90 \quad 110 \quad 33 \quad 30 \quad 34)$$

$$\tilde{f} = \begin{bmatrix} - & 0.59 & 0.58 & 0.6 & - & - & - & - \\ - & - & - & - & - & 1.94 & - & - \\ - & - & - & - & - & - & 0.63 & - \\ - & - & - & - & 0.58 & 0.72 & - & - \\ - & - & - & - & - & - & - & 1.99 \\ - & - & - & - & - & - & 0.72 & 2.72 \\ - & - & - & - & - & - & - & - \\ - & - & - & - & - & - & - & - \end{bmatrix}$$

With respect to these matrices, the Steiner tree model is:

This model is performed using MOACO in 500 iterations each having 60 ants in MATLAB 9 software. The Pareto solutions are obtained and shown in Table 3. Some of the Pareto solutions are shown in figures 12-16. Fig. 11 shows the Pareto solutions in three-dimensional (cubic) space resulted from the three objective functions.

In Fig. 11, X-axis shows cost and axes Y and Z are symmetry of benefit and customers' total value, respectively.



$$\begin{aligned}
 \max \quad & X = 1500x_{12} + 1300x_{13} + 100x_{14} + 208x_{26} + 110x_{37} + 50x_{45} + 132x_{46} + 130x_{58} + 84x_{67} + 135x_{68} \\
 \max \quad & X' = 0.59x_{12} + 0.58x_{13} + 0.6x_{14} + 1.94x_{26} + 0.63x_{37} + 0.58x_{45} + 0.72x_{46} + 1.99x_{58} + 0.72x_{67} + 2.72x_{68} \\
 \min \quad & Y = 450y_1 + 660y_2 + 780y_3 + 90y_4 + 110y_5 + 33y_6 + 30y_7 + 34y_8 \\
 \text{s.t.} \quad & \\
 & x_{12} + x_{13} + x_{14} + x_{26} + x_{37} + x_{45} + x_{46} + x_{58} + x_{67} + x_{68} = y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 - 1 \\
 & x_{14} \leq y_1 \\
 & x_{14} \leq y_4 \\
 & x_{26} \leq y_2 \\
 & x_{26} \leq y_6 \\
 & x_{37} \leq y_3 \\
 & x_{37} \leq y_7 \\
 & x_{46} \leq y_4 \\
 & x_{46} \leq y_6 \\
 & x_{58} \leq y_5 \\
 & x_{58} \leq y_8 \\
 & x_{12} \leq y_1 \\
 & x_{12} \leq y_2 \\
 & x_{67} \leq y_6 \\
 & x_{67} \leq y_7 \\
 & x_{13} \leq y_1 \\
 & x_{13} \leq y_3 \\
 & x_{45} \leq y_5 \\
 & x_{45} \leq y_4 \\
 & x_{68} \leq y_6 \\
 & x_{68} \leq y_8 \\
 & x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_2 + y_4 \\
 & x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_2 + y_6 \\
 & x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_6 + y_4 \\
 & x_{14} + x_{26} + x_{12} + x_{46} \leq y_6 + y_2 + y_4 \\
 & x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_5 + y_6 \\
 & x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_5 + y_8 \\
 & x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_8 + y_6 \\
 & x_{45} + x_{58} + x_{68} + x_{46} \leq y_8 + y_5 + y_6 \\
 & x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_3 + y_6 \\
 & x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_3 + y_7 \\
 & x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_7 + y_6 \\
 & x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_7 + y_3 + y_6 \\
 & x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_7 + y_2 + y_3 + y_6
 \end{aligned}$$



$$x_{14} + x_{46} + x_{67} + x_{37} + x_{13} \leq y_1 + y_4 + y_3 + y_6$$

$$x_{14} + x_{46} + x_{67} + x_{37} + x_{13} \leq y_1 + y_4 + y_3 + y_7$$

$$x_{14} + x_{46} + x_{67} + x_{37} + x_{13} \leq y_1 + y_4 + y_7 + y_6$$

$$x_{14} + x_{46} + x_{67} + x_{37} + x_{13} \leq y_1 + y_7 + y_3 + y_6$$

$$x_{14} + x_{46} + x_{67} + x_{37} + x_{13} \leq y_7 + y_4 + y_3 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_1 + y_2 + y_4 + y_5 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_1 + y_2 + y_4 + y_5 + y_8$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_1 + y_2 + y_4 + y_8 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_1 + y_2 + y_8 + y_5 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_1 + y_8 + y_4 + y_5 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68} \leq y_8 + y_2 + y_4 + y_5 + y_6$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_3 + y_4 + y_5 + y_6 + y_7$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_3 + y_4 + y_5 + y_6 + y_8$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_3 + y_4 + y_5 + y_8 + y_7$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_3 + y_4 + y_8 + y_6 + y_7$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_3 + y_8 + y_5 + y_6 + y_7$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_1 + y_8 + y_4 + y_5 + y_6 + y_7$$

$$x_{14} + x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13} \leq y_8 + y_3 + y_4 + y_5 + y_6 + y_7$$

$$y_h = 1, \quad \forall h \in \{1, 2, 3\}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j$$

$$y_i \in \{0, 1\}, \quad \forall i.$$

Table 3  
The Pareto solutions resulted from the three objective functions

The number of the Pareto solution	cost	benefit	customers total value
1	1957	3143	5.83
2	1890	2800	1.17
3	1987	3227	6.55
4	2187	899	9.32
5	2187	3515	9.17
6	1987	2307	6.63
7	2067	3273	7.82
8	2097	2167	8.63
9	2157	3405	8.54
10	2097	3383	8.54
11	2077	3385	7.18
12	2047	3275	6.55
13	1920	2910	1.80
14	1953	1902	3.88
15	1953	3118	3.74
16	1923	3008	3.11
17	2187	3489	9.26
18	2187	2299	9.31



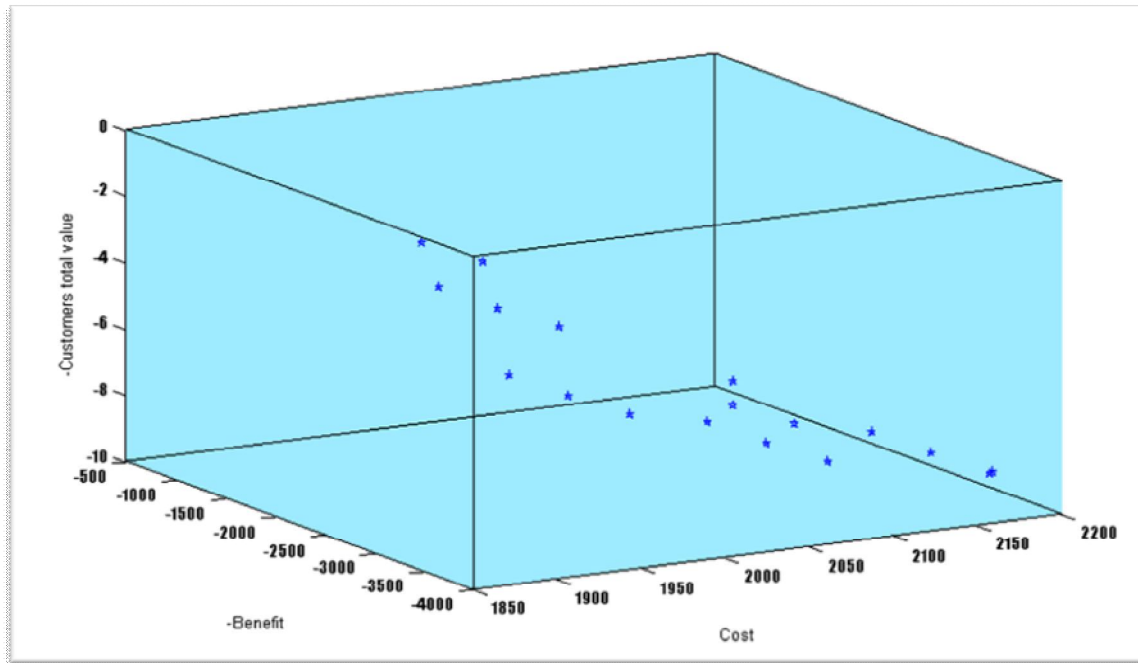


Fig. 11. The Pareto solutions in three-dimensional (cubic) space resulted from the three objective functions

- 1) The first Pareto optimal solution is  $Y^* = 1920$ ,  $X^* = 2910$  and  $X'^* = 1.8$ , with the optimal network as shown in Fig. 12.

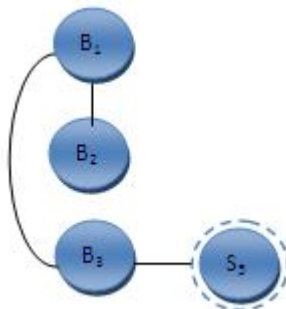


Fig. 12. The first Pareto optimal solution

- 2) The second Pareto optimal solution is  $Y^* = 1923$ ,  $X^* = 3008$  and  $X'^* = 3.11$ , with the optimal network as shown in Fig. 13.

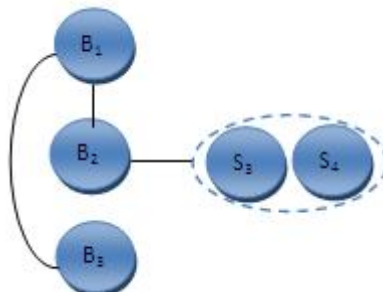


Fig. 13. The second Pareto optimal solution

- 3) The third Pareto optimal solution is  $Y^* = 1957$ ,  $X^* = 3143$  and  $X'^* = 5.83$ , with the optimal network as shown in Fig. 14.



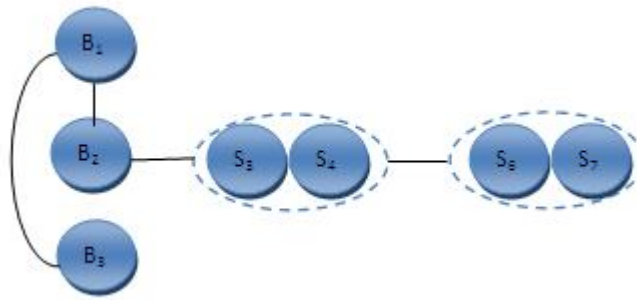


Fig. 14. The third Pareto optimal solution

- 4) The forth Pareto optimal solution is  $Y^* = 2067$ ,  $X^* = 3273$  and  $X'^* = 7.82$ , with the optimal network as shown in Fig. 15.

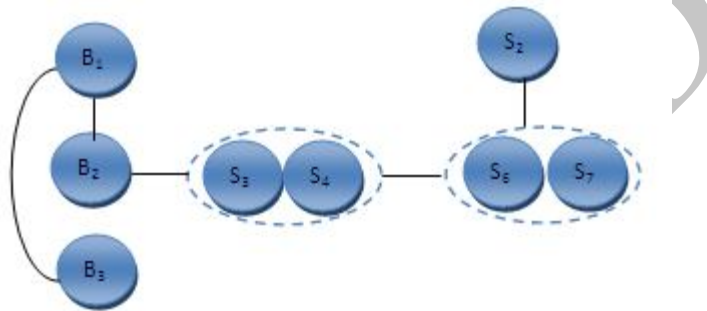


Fig. 15. The forth Pareto optimal solution

- 5) The final Pareto optimal solution is  $Y^* = 2187$ ,  $X^* = 899$  and  $X'^* = 9.32$ , with the optimal network as shown in Fig. 16.

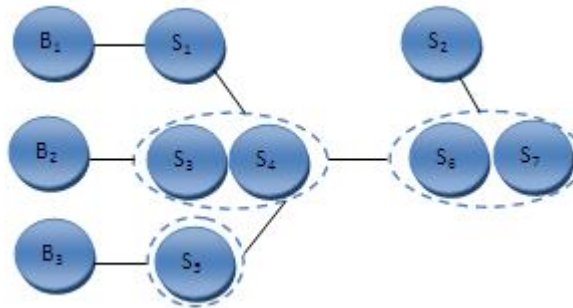


Fig. 16. The final Pareto optimal solution

As shown in figures 12-16, the proposed method provides different products for producers and consumers having different benefits, costs and customers' values. The numerical results imply the configuration of different products having various costs and customers' values being based on customers' views obtained from the web based system. The products themselves are the ones providing maximum benefits for the producers. The significant decision made in the proposed methodology is the trade off between the cost, benefit and customer's value objectives which is based on customers' views on adding features of products and producers' views on configuration of beneficial features.

## 5. Performance on Medium to Large Size Problems

To evaluate the performance of the proposed MOACO, we developed 10 test problems in medium and large sizes with the number of nodes, edges and terminals as given in Table 4.

To evaluate the performance of the proposed MOACO, two useful metrics in the literature were taken into account. These metrics are described below.

- **Quality Metric (QM):** This metric was proposed by Schaffer (1985) and some other researchers. It represents the number of Pareto-optimal solutions found by algorithm.



• **Diversity Metric (DM):** This metric was applied by Zitzler et al. (1999) and was used to measure the spread of the solutions in the final obtained Pareto solutions set found by algorithm. This metric is calculated by

$$D = \sqrt{\sum_{i=1}^n \max(|x_i - y_i|, x, y \in F)}, \quad (24)$$

where  $F$  represents the set of obtained Pareto solutions,  $x$  and  $y$  are two solution vectors of Pareto frontier,  $n$  represents the dimension of the solution space which is equal to the number of objective functions. For our work here,  $n$  is equal to 3.

The computational results corresponding to these two performance metrics for medium to large size problems are presented in Tables 5-6. All test problems have executed 10 times and computational results are presented in Tables 5-6.

Table 7 show that the computational time needed by MOACO.

As shown in Tables 5 and 6, by increasing the problem sizes the number of Pareto solutions is also rises drastically showing the efficiency of the MOACO algorithm. Also, according to Table 7 the solutions are obtained in reasonable time.

Table 4  
Graph sizes

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
$ V $	8	12	15	17	18	20	23	30	35	40
$ E $	10	16	20	23	25	30	40	47	53	62
$ N $	3	4	5	6	7	8	9	14	15	15

Table 5  
Computational results for the quality metric

Test performance	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
1	18	25	35	48	52	67	91	145	215	353
2	18	25	33	45	52	69	84	132	195	310
3	18	25	37	47	52	69	96	122	220	327
4	18	25	35	46	50	68	87	149	203	278
5	18	25	35	46	55	69	62	145	187	312
6	18	25	34	47	52	67	93	118	211	351
7	18	25	35	46	54	62	85	136	219	333
8	18	25	36	45	52	59	74	134	220	275
9	18	25	37	43	52	67	94	121	218	363
10	18	25	35	48	53	62	72	116	184	298

Table 6  
Computational results for the diversity metric

Test performance	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
1	54.04	63.64	72.56	78.12	80.38	91.70	107.58	128.45	148.37	191.48
2	54.04	63.64	72.56	78.12	80.38	91.70	105.90	128.45	145.60	184.71
3	54.04	63.64	72.56	78.12	80.38	91.70	108.61	123.80	148.37	191.48
4	54.04	63.64	72.56	78.12	80.38	91.70	107.58	128.45	148.37	184.71
5	54.04	63.64	72.56	78.12	80.38	91.70	93.60	128.45	141.33	184.71
6	54.04	63.64	72.56	78.12	80.38	91.70	107.58	120.62	147.41	191.48
7	54.04	63.64	72.56	78.12	80.38	89.28	105.90	128.45	148.37	191.48
8	54.04	63.64	72.56	78.12	80.38	89.28	99.36	128.45	148.37	188.39
9	54.04	63.64	72.56	77.51	80.38	91.70	107.58	122.23	148.37	191.48
10	54.04	63.64	72.56	78.12	80.38	89.28	105.90	123.80	146.50	185.26

Table 7  
The mean values of computing times (seconds) for MOACO

Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
6	14	45	65	69	91	102	235	290	385

## 6. Conclusions

We proposed an ant colony optimization to determine value adding functionalities for convergent products. A collection of base functions and sub-functions configured the nodes of a web-based (digital) network representing

functionalities. Each arc in the network was to be assigned as the link between two nodes. The aim was to find an optimal tree of functionalities in the network adding value to the product in the web environment. First,



a purification process was performed in the product network to assign the links among bases and sub-functions. Then, numerical values as benefits and costs were determined for arcs and nodes, respectively, using leveling and clustering approaches. Finally, the Steiner tree methodology was adapted to a multi-objective model of the network to find the optimal tree determining the value adding sub-functions to bases in a convergent product. The numerical results can be used for the configuration of different products having various costs based on customer's view obtained from the web based system.

## 7. Acknowledgements

The first two authors thank Mazandaran University of Science and Technology and the third author thanks Sharif University of Technology for supporting this work.

## References

- [1] Baran B., Schaerer M., (2003). A multiobjective ant colony system for vehicle routing problem with time windows, in: Proceedings of the Twenty First IASTED International Conference on Applied Informatics, February, 97–102.
- [2] Beasley J. E., (1989). An SST-based algorithm for the Steiner problem in graphs. *Networks*, 19(1), 1–16.
- [3] Callahan J., Lasry E. (2004). The importance of customer input in the development of very new products. *R&D Management*, 34(2), 107–120.
- [4] Cardoso P., Jesus M., Marquez A., (2003). MONACO-multi-objective network optimization based on an ACO, in: Proceedings of the X Encuentros de Geometria Computacional, Seville, Spain, 16–17.
- [5] Chakrabarty D., Könemann J., Pritchard D., (2010). Integrality gap of the hypergraphic relaxation of Steiner trees: A short proof of a 1.55 upper bound. *Operations Research Letters*, 38 (6), 567–570.
- [6] Chimani M., Mutzel P., Zey B., (2012). Improved Steiner tree algorithms for bounded treewidth. *Journal of Discrete Algorithms*, 16, 67–78.
- [7] Colomi A., Dorigo M., Maniezzo V., Trubian M., (1994). Ant system for job-shop scheduling, *Belgian Journal of Operations Research, Statistics and Computer Science* 34, 295–305.
- [8] Costa A. M., Cordeau J. F., & Laporte G., (2006). Steiner tree problems with profits. *INFOR* 44(2), 99–115.
- [9] Doerner K., Gutjahr W.J., Hartl R.F., Strauss C., (2004). Stummer, C., Pareto ant colony optimization: a meta-heuristic approach to multiobjective portfolio selection, *Annals of Operation Research* 131(1–4), 79–99.
- [10] Doerner K., Hartl R.F., Teimann M., (2003). Are COMPETants more competent for problem solving?—the case of full truckload transportation, *Central European Journal of Operation Research*, 11(2) 115–141.
- [11] Dorigo M., Caro G.D., Gambardella L.M., (1999). Ant algorithms for discrete optimization, *Artificial Life* 5 (2), 137–172.
- [12] Dorigo M., Gambardella L.M., (1997). A cooperative learning approach to the traveling salesman problem, *IEEE Transaction on Evolutionary Computation* 1, 53–66.
- [13] Dorigo M., Gambardella L.M., (1997). Ant Colonies for the Traveling Salesman Problem, *BioSystems* 43, 73–81.
- [14] Dorigo M., Maniezzo V., Colomi A., (1991). Ant System: An Autocatalytic Optimizing Process, Technical Report, Dipartimento di Elettronica e Informazione, Politecnico di Milano.
- [15] Dorigo M., Maniezzo V., Colomi A., (1996). Ant System: Optimization by a Colony of Cooperative Agents, *IEEE Transactions on Systems, Man and Cybernetics, Part B* 26(1), 29–41.
- [16] Dreyfuss S.E., Wagner R.A., (1971). The Steiner problem in graphs. *Networks* 1(3), 195–207.
- [17] Duin C.W., Volgenant A., (1989). Reduction tests for the Steiner problem in graphs, *Networks*, 19(5), 549–567.
- [18] Esbensen H., (1995). Computing near-optimal solutions to the Steiner problem in a graph using a genetic algorithm. *Networks*, 26(4), 173–185.
- [19] Fatta G. Di, Presti G. Lo, Re G. Lo, (2003). A parallel genetic algorithm for the Steiner problem in networks, 15th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2003), Marina del Rey, CA, USA, 569–573.
- [20] Gambardella L.M., Dorigo M., (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing* 12, 237–255.
- [21] Gamzu I., Segev D., (2010). A polylogarithmic approximation for computing non-metric terminal Steiner tree. *Information Processing Letters*, 110(18–19), 826–829.
- [22] Garcia-Martinez C., Cordon O., Herrera F., (2007). A taxonomy and empirical analysis of multiple objective ant colony optimization algorithm for the bi-criteria TSP, *European Journal of Operation Research* 180, 116–148.
- [23] Gill T. (2008). Convergent products: What functionalities add more value to the base? *Journal of Marketing*, 72(2), 46–62.
- [24] Gill T., Lei J. (2009). Convergence in the high-technology consumer markets: Not all brands gain equally from adding new functionalities to products. *Marketing Letters*, 20(1), 91–103.
- [25] Gosavi S., Das S., Vaze S., Singh G., Buehler E., (2003). Obtaining Subtrees from Graphs: An Ant Colony Approach. In Proceedings of the IEEE Swarm Intelligence Symposium, 160–166.
- [26] Hakimi S.B., (1971). Steiner's problem in graphs and its implications. *Networks*, 1, 113–133.
- [27] Hwang F.K., Richards D.S., (1992). Steiner tree problems. *Networks*, 22(1), 55–89.
- [28] Iredi S., Merkle D., Middendorf M., (2001). Bi-criterion optimization with multi colony ant algorithm, in: Proceedings of the First International Conference on Evolutionary Multi-criteria Optimization (EMO) 359–372.
- [29] Jangam S.R., Chakraborti N., (2007). A novel method for alignment of two nucleic acid sequences using ant colony optimization and genetic algorithm. *Applied Soft Computing* 7, 1121–1130.
- [30] Johnson D.S., (1985). The NP-completeness column: An ongoing guide. *Journal of Algorithms*, 6(3), 434–451.
- [31] Kapsalis A., Rayward-Smith V.J., Smith G.D., (1993). Solving the graphical Steiner tree problem using genetic



- algorithms. *Journal of the Operational Research Society*, 44(4), 397–406.
- [32] Koch T., Martin A., (1998). Solving Steiner tree problems in graphs to optimality. *Networks*, 32(3), 207–232.
- [33] Kou L., Markowsky G., Berman L., (1981). A fast algorithm for Steiner trees. *Acta Informatica*, 15(2), 141–145.
- [34] Lawler E. L., (1976). *Combinatorial Optimization Networks and Matroids*. Holt, Rinehart and Winston, New York.
- [35] Lee C., Song B., Park Y., (2012). Design of convergent product concepts based on functionality: An association rule mining and decision tree approach. *Expert Systems with Applications*, 39, 9534–9542.
- [36] Lu X.x., Yang S.w., Zheng N., (2011). Location-selection of Wireless Network Based on Restricted Steiner Tree Algorithm. *Procedia Environmental Sciences*, 10(A), 368–373.
- [37] Martins S.L., Pardalos P., Resende M.G., Ribeiro C.C., (1999). Greedy randomized adaptive search procedures for the Steiner problem in graphs. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 43, 133–146.
- [38] Martins S.L., Resende M.G.C., Ribeiro C.C., Pardalos P.M., (2000). A parallel GRASP for the Steiner tree problem in graphs using a hybrid local search strategy. *Journal of Global Optimization*, 17(1), 267–283.
- [39] Martins S.L., Ribeiro C.C., Souza M.C., (1998). A parallel GRASP for the Steiner problem in graphs. *Lecture Notes in Computer Science*, Springer-Verlag, 1457, 310–331.
- [40] Mehlhorn K., (1988). A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters Archive*, 27(3), 125–128.
- [41] Müller-Hannemann M., Tazari S., (2010). A near linear time approximation scheme for Steiner tree among obstacles in the plane. *Computational Geometry*, 43(4), 395–409.
- [42] Rayward-Smith V.J., Clare A., (1986). On finding Steiner vertices. *Networks*, 16(3), 283–294.
- [43] Ribeiro C.C., Souza M.C., (2000). Tabu search for the Steiner problem in graphs. *Networks*, 36(2), 138–146.
- [44] Robins G., Zelikovsky A., (2000). Improved Steiner tree approximation in graphs. in *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, Philadelphia, ACM, New York, 770–779.
- [45] Robins G., Zelikovsky A., (2005). Tighter bounds for graph Steiner tree approximation. *SIAM Journal on Discrete Mathematics*, 19 (1), 122–134.
- [46] Schaffer J.D., (1985). Multiple objective optimizations with vector evaluated genetic algorithms, in: J.D. Schaffer (Ed.), *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Lawrence Erlbaum, Hillsdale, New Jersey, 93–100.
- [47] Socha K., Sampels M., Manfrin M., (2003). Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. in: *Proceedings of the Third European Workshop on Evolutionary Computation in Combinatorial Optimization (EvoCOP)* 334–345.
- [48] Stützle T., Dorigo M., (1999). *ACO Algorithms for the Traveling Salesman Problem*. Working Paper, Université Libre de Bruxelles, Belgium.
- [49] Takahashi H., Matsuyama A., (1980). An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24(6), 573–577.
- [50] Verworn B., Herstatt C., Nagahira A. (2008). The fuzzy front end of Japanese new product development projects: Impact on success and differences between incremental and radical projects. *R&D Management*, 38(1), 1-19.
- [51] Winter P., MacGregor Smith J., (1992). Path-distance heuristics for the Steiner problem in undirected networks. *Algorithmica*, 7(2&3), 309-327.
- [52] Yoffie D. B. (1997). Introduction: Chess and competing in the age of digital convergence. In D. B. Yoffie (Ed.), *Competing in the age of digital convergence* (pp. 1–36). Boston: Harvard Business School Press.
- [53] Zitzler E., Thiele L., (1999). *Evolutionary Algorithm for Multi-Objective Optimization: Methods and Applications*, Ph.D. Dissertation, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.
- [54] Zych A., Bilò D., (2011). New Reoptimization Techniques applied to Steiner Tree Problem. *Electronic Notes in Discrete Mathematics*, 37, 387-392.