



Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
www.scientiairanica.com



An ε -constraint multi-objective optimization model for web-based convergent product networks using the Steiner tree

R. Hassanzadeh^{a,*}, I. Mahdavi^a and N. Mahdavi-Amiri^b

a. *Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran.*

b. *Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran.*

Received 7 December 2012; received in revised form 11 June 2014; accepted 7 July 2014

KEYWORDS

Convergent product;
Web-based (digital)
network;
Multi-objective
programming;
Steiner tree;
 ε -constraint.

Abstract. A 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 value corresponding to the benefits is considered. Finally, the Steiner tree methodology is adapted to a multi-objective model optimized by an augmented ε -constraint method. An example is worked out to illustrate the proposed approach.

© 2015 Sharif University of Technology. All rights reserved.

1. Introduction

Convergence in electronic and communication sectors has enabled the addition of disparate new functionalities to existing base functions (e.g., adding mobile televisions 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.

A convergent product is similar to product assembly, where different parts of a product combine to configure a final product. Thus, the designer (modeler) of an 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 the strategic gravity of convergent products that are formed by adding new functions to an existing base product [1]; multiple functions are integrated together in one device, rather than delivered separately, to work better. Representative examples of this shifting trend include the Apple iPhone and the Microsoft Xbox. Such convergent products have created new business opportunities for companies to gain or maintain a competitive edge, bringing about

*. *Corresponding author. Tel.: +98 191 3237455*
E-mail addresses: rhz_1974@ustmb.ac.ir (R. Hassanzadeh);
irajarash@rediffmail.com (I. Mahdavi); nezamm@sharif.edu
(N. Mahdavi-Amiri)

immense changes in a wide array of industries [2]. Consequently, design of Convergent Product Concepts (CPCs) has likewise become an integral part of business concerns [3]. This is of particular importance in today's business environment, where markets shift rapidly, technologies proliferate unceasingly, and thus business life cycles tend to be shorter.

The systematic design of a CPC needs to address the following analytic issues; first, being the type of data to be employed. The concept design aims to incorporate customer need into the design specs [4]. A deeper understanding of the fuzzy front end could help firms to be more successful in their efforts to develop new products [5]. Lee et al. [6] proposed a systematic approach for the design of CPCs based on online community information using data mining techniques.

For instance, the ability of the assembly modeler to furnish information regarding 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 interference on the shop floor. Also, the 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 the common Design For Assembly (DFA) guidelines are present, such as the 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 analyses. 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 the acceptable tolerances of 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 a 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 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 subset $N = V - N$ of the *Steiner nodes*. The Steiner tree problem is NP-hard for most relevant classes of graph (see [7]).

The Steiner problem in graphs was originally formulated by Hakimi [8]. Since then, the problem has received considerable attention in the literature. Several exact algorithms and heuristics have been proposed and discussed. Hakimi remarked that a *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 [9] 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 problems was given by Hwang and Richards [10], Alvarez-Miranda et al. [11]

and Fu and Hao [12]. Several exact algorithms have been proposed, such as the dynamic programming technique by Dreyfuss and Wagner [13], Lagrangean relaxation approach by Beasley [14] and branch-and-cut algorithm by Koch and Martin [15]. Duin and Volgenant [16] presented some techniques to reduce the size of the graphs for the Graphical Steiner tree Problem (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 the Shortest Path Heuristic (SPH) by Takahashi and Matsuyama [17], the Distance Network Heuristic (DNH) by Kou et al. [18], the Average Distance Heuristic (ADH) by Rayward-Smith and Clare [19] and the Path-Distance Heuristic (PDH) by Winter and MacGregor Smith [20]. Mehlhorn [21] modified DNH to arrive at a more efficient algorithm. Robins and Zelikovsky [22,23] proposed algorithms improving the performance ratio.

Recently, metaheuristics have been considered to propose better methods for finding near optimal solutions. Examples are the Genetic Algorithm (GA) [24,25], GRASP [26] and Tabu search [27]. Although these algorithms have polynomial time complexities, in general, they are enormously costly on large scale input sets. To deal with the cost issue, some parallel metaheuristic algorithms have been proposed, such as parallel GRASP [28], parallel GRASP using a hybrid local search [29] and parallel GA [30].

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

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 a solution algorithm. Section 4 works out an experimental study to illustrate the proposed algorithm. Section 5 concludes the paper.

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, the customer clicks the “classify” button and observes the classified functionalities in a separate web page. This process is shown in Figure 1.

The aim in designing such a web based system is to obtain the configuration of products having functions and promoting or adding the corresponding subfunctions as value added purposes of sale.

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.

Mathematical notations

i and j	Index for functions and sub-functions, $i = 1, \dots, m$ and $j = m + 1, \dots, m + n$;
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-dimensional comparison matrix, F , is shown in Figure 2. Note that customers fill in this matrix using numerical values, $F_{ijk} \in [0, 1]$.

This matrix is normalized to remove the scales. The normalized values are shown by F_{ijk}^{norm} . A threshold value of θ is considered in such a way that $f_{ij} = \frac{\sum_{k=1}^p F_{ijk}^{\text{norm}}}{p} \geq \theta$ are chosen to be assigned as links. f_{ij} is a value that customers consider for $\text{arc}(i, j)$. These links configure a network called a purified network, as shown in Figure 3.

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. Here, Algorithm 1 is proposed.

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.

Algorithm 1. Leveling to configure a leveled network.

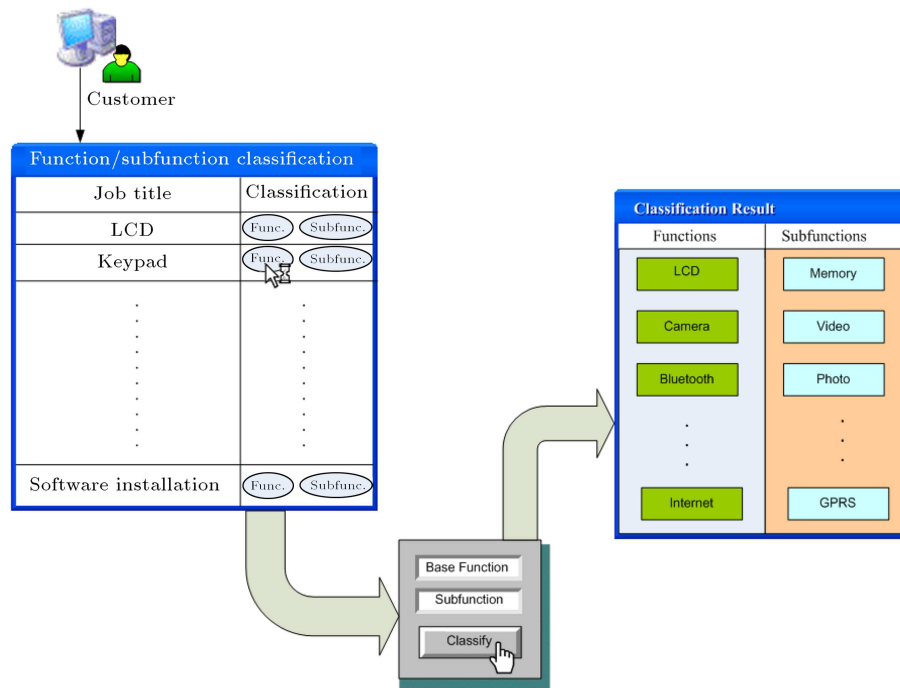


Figure 1. Classification process.

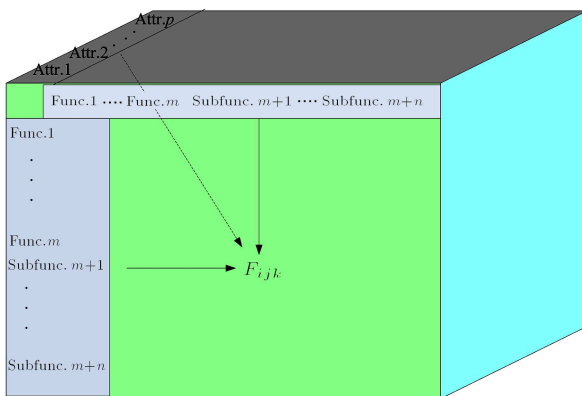


Figure 2. Three-dimensional comparison matrix.

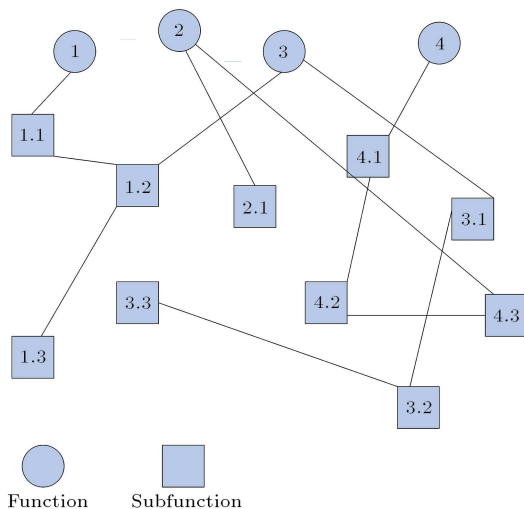


Figure 3. A purified network.

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 are grouped in a cluster. Therefore, clusters consisting of different nodes are configured. These clusters are being configured as a new network. The leveling and clustering processes are shown schematically in Figure 4. Later, we apply the Steiner tree methodology to optimize this network.

Algorithm 2 is proposed for clustering. 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 a 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

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

End for.

Step 2: Stop.

Algorithm 2. Clustering of levels in a leveled network.

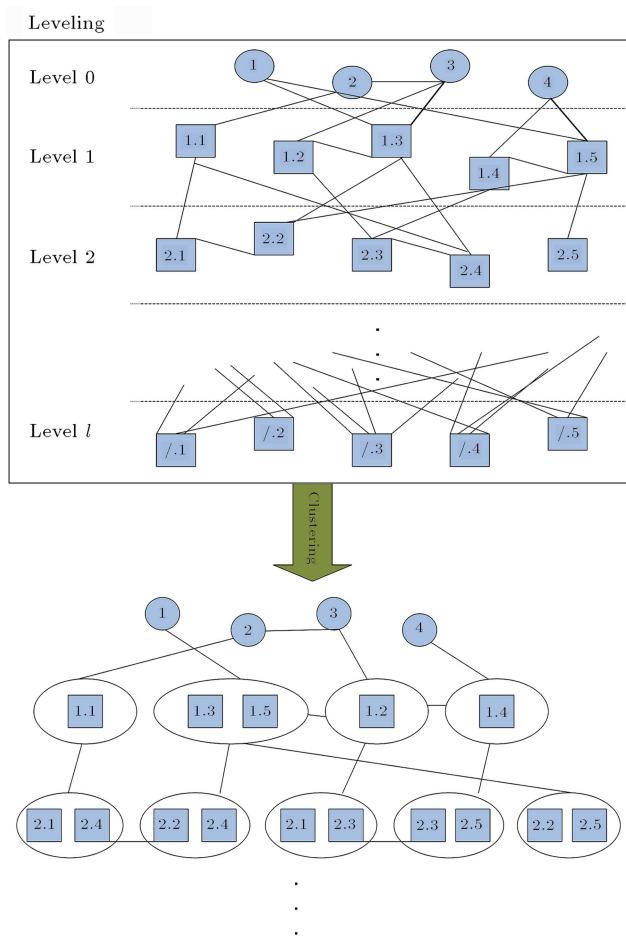


Figure 4. Leveling and clustering processes.

minimize cost, and maximize benefit and customer 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:

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, using the Steiner tree in the proposed network, and the NP-hardness of the problem, we use 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 :

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

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

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

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);

α_{ij} : The software reduction cost coefficient of node i in cluster j ;

β_{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:

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

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 ;

α_{ij} : The software reduction cost coefficient of node i in cluster j ;

β_{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:

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

where:

$p_{jj'}$: 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 customer total value in clusters, the following equation is used:

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

where:

- $f_{jj'}$: The adjusted arc customer 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 1. The node having a minimal cost is chosen for level 1. Then, instead of using the zero/one mathematical program for level 1, we can use Step 3 of Algorithm 3. This leads to a reduction of computations by avoiding the need to use the zero/one programs.

3. Mathematical formulation and solution method

Here, first, the mathematical model for the considered problem is proposed and then the solution approach is stated.

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**
 $\{\text{Form clusters at level } i\}$
 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 $z_{kj} = 0, \forall j \in m_{ik}, i \neq p$;
 $n_i \leftarrow n_i - \{k\}$.
 End while;
 For $j = 1$ till q_i , **do**
 $\{q_i$ is the number of clusters in 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.**

Algorithm 3. Leveling and clustering in the network.

3.1. Mathematical formulation

We first recall the undirected Dantzig-Fulkerson-Johnson model for the Convergent Product Steiner Tree Problem (CPSTP) proposed in [31]. 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:

$$\text{Maximize} \quad \sum_{(i,j) \in E} p_{ij} \cdot x_{ij}, \quad (1)$$

$$\text{Maximize} \quad \sum_{(i,j) \in E} f_{ij} \cdot x_{ij}, \quad (2)$$

$$\text{Minimize} \quad \sum_{i \in V} c_i, y_i. \quad (3)$$

such that:

$$\sum_{(i,j) \in E} x_{ij} = \sum_{i \in V} y_i - 1, \quad (4)$$

$$\sum_{(i,j) \in E(S)} x_{ij} \leq \sum_{i \in S - \{k\}} y_i, \quad \forall k \in S \subseteq V, \quad \forall S : |S| \geq 2, \quad (5)$$

$$y_h = 1, \quad \forall h \in N, \quad (6)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in E, \quad (7)$$

$$y_i \in \{0, 1\}, \quad \forall i \in V. \quad (8)$$

The objectives are to maximize the aggregated benefits, minimize the aggregated costs and maximize the aggregated customer total value. Constraint (4) guarantees that the number of clusters in a solution is equal to the number of links minus one, and Constraints (5) are the connectivity constraints. The number of constraints in (5) 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 (6) impose the terminal clusters to exist in the tree. Relations (7) and (8) show the variable types.

3.2. Multi-objective solution method

Decision making issues can rarely rely on a single well defined criterion. Although the multiple facets of a decision process can be aggregated into a single objective function, this simplification involves arbitrary rules that can hardly adequately capture the complexity of real world decision making issues. Thereby, the

interest in multi-criteria decision making has continually grown during the past decades, as attested by the number of books and surveys on the topic (e.g., see [32–35], among others). It comes as no surprise that more and more publications address combinatorial issues, given that many real world applications involve discrete decisions or events. The reader is referred to [36] for a review of the literature on Multi-Objective Combinatorial Optimization (MOCO) problems. Among exact methods to find the Pareto front of MOCO problems, weighted sum scalarization is the most popular according to [36]. This method solves different single objective subproblems generated by a linear scalarization of the objectives. By varying the weights of this linear function, all supported non-dominated points can be found. On the other hand, linear scalarization cannot find unsupported points and is, therefore, ill-suited for non-convex objective spaces, such as those associated with MOCO problems. This drawback can be overcome with the two-phase method of [37] that finds all supported points through a weighted sum scalarization in the first phase, while non-supported points are found during the second phase with problem specific methods. Most algorithms that find the exact Pareto front of MOCO problems are variants of the two-phase method of [36], although other parametric approaches, based on weighted scalarizations, can find the exact Pareto front of Bi-Objective Combinatorial Optimization (BOCO) problems [38–40]. Besides weighting sum algorithms, the ε -constraint method [32,34] is the best known approach for solving MOCO problems, according to [36]. This method generates single objective subproblems, called ε -constraint problems, by transforming all but one objective into constraints. The upper bounds of these constraints are given by the ε -vector and, by varying it, the exact Pareto front can, theoretically, be generated. In practice, because of the high number of subproblems and the difficulty in establishing an efficient variation scheme for the ε -vector, this approach has mostly been integrated within heuristic and interactive schemes. It can, however, generate the exact Pareto front in particular situations.

3.2.1. The ordinary ε -constraint method

Consider a MOCO problem with k objective functions, $f_i(x), i = 1, \dots, k$, where $x \in X$ is the vector of decision variables, and X is the feasible solution space determined by the constraints of the MOCO problem. Without loss of generality, we assume here that all objective functions are for minimization. In the ε -constraint method, one of the objective functions, as the main objective function, is optimized using the other objective functions added as constraints to the feasible solution space of X , as follows:

$$\text{Min } f_1(x)$$

such that:

$$f_2(x) \leq \varepsilon_2, f_3(x) \leq \varepsilon_3, \dots, f_k(x) \leq \varepsilon_k, \quad x \in X. \quad (9)$$

By parametric variation of the right hand side $(\varepsilon_2, \dots, \varepsilon_k)$ of the newly added constraints, the Pareto optimal solutions are obtained. In order to apply the ε -constraint method, the range of at least $p - 1$ objective functions is needed to determine grid points for the $\varepsilon_2, \dots, \varepsilon_k$ values. The most common approach is to calculate these ranges from the payoff table by individually optimizing each objective function. The mathematical details of computing the payoff table for a MOCO problem can be found in [41]. The minimum and maximum values of the i th objective function are individually calculated using the payoff table and denoted by $f_i^{\min}$ and $f_i^{\max}$, respectively. Then, the range of the i th objective function is determined as follows:

$$r_i = f_i^{\min} - f_i^{\max}.$$

The range r_i is divided into q_i equal intervals. Then, ε_i in Relation (1) is set to these $q_i + 1$ grid points by:

$$\varepsilon_i^p = f_i^{\max} - \frac{r_i}{q_i} \times p, \quad p = 0, \dots, q_i, \quad i = 2, \dots, k,$$

where p is the grid point number. Using the ε -constraint method, we indeed convert the MOCO problem into $\prod_{i=2}^k (q_i + 1)$ single objective optimization sub-problems. Each sub-problem has the solution space, X , which is further limited by its own inequality constraints for $f_2, \dots, f_k$. Each sub-problem results in a candidate solution for the MOCO problem or a Pareto optimal solution. At the same time, some sub-problems may become infeasible (the solution space may be empty) due to the added constraints introduced by $f_2, \dots, f_k$; such sub-problems are discarded. A decision maker is then used to select the most preferred solution out of the obtained Pareto optimal solutions.

3.2.2. The augmented ε -constraint method

The drawback of the ordinary ε -constraint method is the efficiency of its Pareto solutions. In other words, there is no guarantee for the solutions of the ordinary ε -constraint method to be efficient, and inefficient solutions may be generated. If there is another Pareto solution that can improve at least one objective function without deteriorating the other objective functions, the obtained solution is said to be inefficient. In view of the fact that all the objective functions are supposed to be minimized here, without loss of generality, a greater value for objective functions is more desirable. In fact, the ordinary ε -constraint method mostly gives

solutions as $f_2 = \varepsilon_2, \dots, f_k = \varepsilon_k$. Although all constraints of Relation (9) are satisfied, the solutions may be inefficient. In order to overcome this drawback using the augmented ε -constraint method, at first, the inequality constraints of the objective functions in Eq. (9) are transformed into equality constraints by introducing positive surplus variables, known also as slack variables [42]. Then, the main objective function is augmented by the sum of the surplus values. So, the augmented ε -constraint problem can be formulated as:

$$\text{Min } f_1(x) - \delta \times (s_2 + s_3 + \dots + s_k),$$

such that:

$$f_2(x) = \varepsilon_2 - s_2, f_3(x) = \varepsilon_3 - s_3, \dots, f_k(x) = \varepsilon_k - s_k,$$

$$x \in X, s_i \geq 0, i = 1, \dots, k, \quad (10)$$

where δ is a small number usually between 10^{-3} and 10^{-6} [43]. The equality constraints of the objective functions in Eq. (10) are equivalent to the inequality constraints in Eq. (9). However, in the ordinary ε -constraint method, the slack variables, s_i , are mostly set to zero by generating solutions as $f_i = \varepsilon_i$. On the other hand, in the augmented ε -constraint method, the main objective function is augmented to include the sum of the slack variables. This mechanism prevents generation of inefficient solutions. It can mathematically be proven that the augmented ε -constraint method only generates efficient solutions. The proof can be found in [43].

Algorithm 4 shows the payoff table determination process using lexicographic optimization, and Algorithm 5 presents an improved augmented ε -constraint [44–46].

A flowchart is added to imply the stages of product development by the convergent product idea in Figure 5.

Step 0. Input model of the multi-objective function.
Step 1. For $i = 1$ till k , do
 Form a single objective optimization problem considering f_i as the objective function: $\{\min f_i(x) \text{ s.t. } x \in X\}$;
 Solve the single objective optimization problem to calculate the diagonal elements of the payoff table $\{F_{ii} = f_i^*(x_i^*)\}$;
 For $j = 1$ till k , do
 If $j = i$, then
 $j = j + 1$
 else
 Form a single objective optimization problem considering f_i as the objective function:
 $\{\min f_i(x) \text{ s.t. } x \in X, f_i = f_j^*(x_j^*)\}$;
 Solve the single objective optimization problem to calculate the off-diagonal elements of the payoff table $\{F_{ij} = f_j^*(x_j^*)\}$;
 End if;
 End for;
 End for.
Step 3. Output the pay-off table.
Step 4. Stop.

Algorithm 4. Determining the pay-off table using lexicographic optimization.

Step 0. Input the pay-off table and q_i for $i = 1, \dots, k$.
Step 1. For $i = 1$ till k , do
 $r_i \leftarrow F_i - f_i^*(x_i^*)$ s.t.
 $F_i = \max\{f_i(x_1^*), f_i(x_2^*), \dots, f_i(x_k^*)\}$;
 For $ni = 0$ till q_i , do
 $e_{i,ni} = F_i - (\frac{r_i}{q_i}) \times ni$;
 End for;
 End for.
Step 2. For $n2 = 0$ till q_2 , do
 For $n3 = 0$ till q_3 , do
 .
 .
 For $nk = 0$ till q_k , do
 Form a single objective optimization sub-problem on the basis of $e_{i,ni}$:
 $\left\{ \begin{array}{l} \min (f_1(x) - r_1 \sum_{i=2}^k (\frac{s_{i,ni}}{r_i})) \\ \text{s.t. } f_i(x) + s_{i,ni} = e_{i,ni}, i = 2, \dots, k \\ x \in X, s_{i,ni} \geq 0 \end{array} \right\}$;
 Solve the single objective optimization problem;
 End for;
 .
 End for;
 End for;
Step 3. Output the Pareto optimal set.
Step 4. Stop.

Algorithm 5. The augmented ε -constraint with lexicographic optimization method.

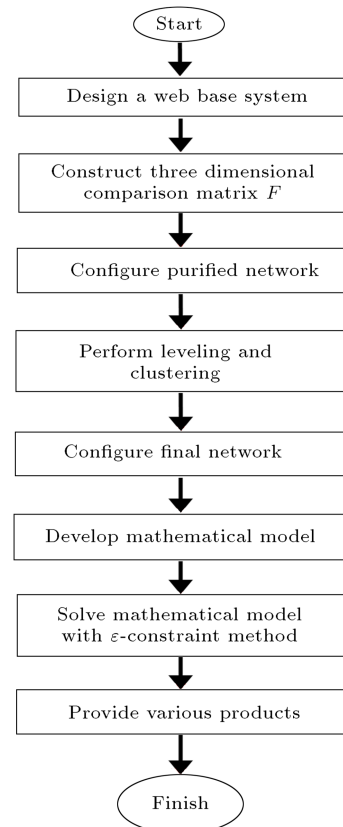


Figure 5. A flowchart of the convergent product process.

4. Experimental study

Here, the case study of a Portable Multimedia Player (PMP) is presented to illustrate the proposed approach. PMP is a typical example of a convergent product which is capable of storing and playing digital

media. Although the initial use of PMP appeared to be in viewing images, and playing music and videos stored in various devices, a variety of functions have now been developed due to the rapid development of information and communication technologies. Representative examples are navigation, Digital Multimedia Broadcasting (DMB), surfing, online communication, and so forth. The afore-designed web base system has been adapted to the PMP device by some web designers. In the proposed web base system, 100 customers were asked to state their opinions of their favorite PMPs. They announced their views by clicking on the options given by the R&D group.

For this case, we consider the internet lecture, an electronic dictionary and surfing as functions or base functions, and battery duration, portability, speed, storage device, audio quality, video quality and input devices as sub-functions. Also, in this example, we consider size, touch screen, speaker type, resolution and RAM as attributes.

For simplicity, we use the notations given in Table 1 to show the convergent product and the proposed network.

The information given by the users as stated above are collected in a three dimensional matrix. The values of attributes for functions and subfunctions are given in Table 2.

Note that the tables related to all three attributes are configured, and their arithmetic means are shown as the final functions, sub-functions and an attribute comparison matrix.

Our threshold value is considered to be 0.561, which is the mean of the data given in Table 2. Therefore, the threshold matrix is shown in Table 3, and the corresponding network is configured as Figure 6.

Table 1. Notations for convergent product in the proposed network.

B_1 = internet lecture	B_2 = electronic dictionary	B_3 = surfing	S_1 = battery duration	S_2 = portability
S_3 = storage device	S_4 = audio quality	S_5 = video quality	S_6 = speed	S_7 = input device

Table 2. Three-dimensional comparison matrix for all the attributes.

Attributes	B_1	B_2	B_3	S_1	S_2	S_3	S_4	S_5	S_6	S_7
B_1	0	0.59	0.58	0.6	0.53	0.66	0.46	0.52	0.48	0.46
B_2	-	0	0.46	0.36	0.46	0.63	0.86	0.58	0.43	0.43
B_3	-	-	0	0.59	0.56	0.53	0.9	0.63	0.33	0.53
S_1	-	-	-	0	0.58	0.59	0.6	0.53	0.6	0.6
S_2	-	-	-	-	0	0.36	0.43	0.43	0.7	0.83
S_3	-	-	-	-	-	0	0.63	0.6	0.63	0.58
S_4	-	-	-	-	-	-	0	0.46	0.6	0.43
S_5	-	-	-	-	-	-	-	0	0.65	0.63
S_6	-	-	-	-	-	-	-	-	0	0.63
S_7	-	-	-	-	-	-	-	-	-	0

Table 3. Threshold comparison matrix for all the attributes.

Attributes	B_1	B_2	B_3	S_1	S_2	S_3	S_4	S_5	S_6	S_7
B_1	0	1	1	1	0	1	0	0	0	0
B_2	-	0	0	0	0	1	1	1	0	0
B_3	-	-	0	1	0	0	1	1	0	0
S_1	-	-	-	0	1	1	1	0	1	1
S_2	-	-	-	-	0	0	0	0	1	1
S_3	-	-	-	-	-	0	1	1	1	1
S_4	-	-	-	-	-	-	0	0	1	0
S_5	-	-	-	-	-	-	-	0	1	1
S_6	-	-	-	-	-	-	-	-	0	1
S_7	-	-	-	-	-	-	-	-	-	0

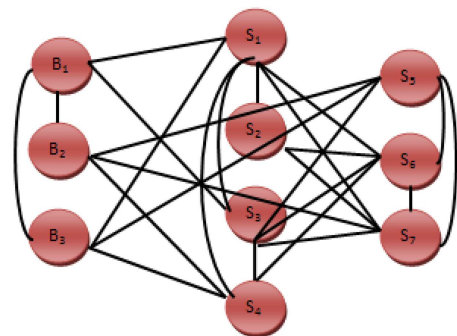


Figure 6. Configured threshold network.

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

The cost vectors, the benefit matrix, customer total value matrix and matrices, $\alpha = [\alpha_{ij}]$, $\beta = [\beta_{ij}]$,

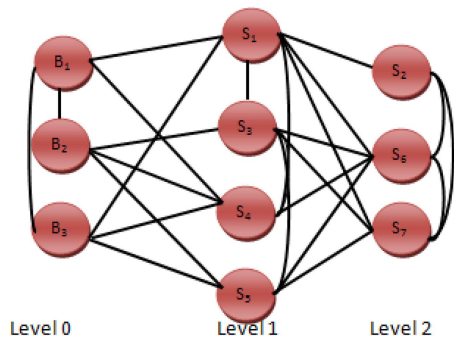


Figure 7. Configured leveled network.

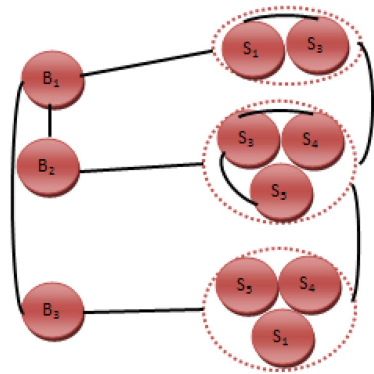


Figure 8. Configured clustered network (second and third steps of Algorithm 3).

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 Figure 9.

In Figure 10, the next iteration of Algorithm 3 for clustering is performed, the purified network is

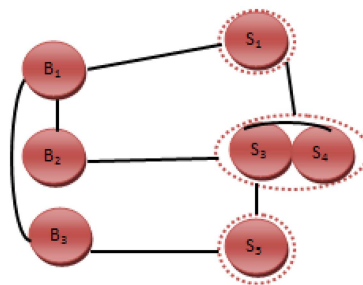


Figure 9. Configured clustered network for level 1.

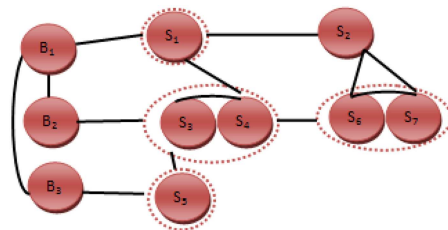


Figure 10. Configured clustered network.

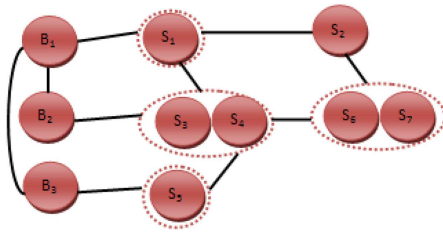


Figure 11. Configured clustered network.

obtained, and the final network is obtained as shown in Figure 11.

After purifying the benefits, costs and customer total values for level 2, the final cost, benefit, and customer 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:

$$\max \quad X = 1500x_{12} + 1300x_{13} + 100x_{14} + 208x_{26}$$

$$+ 110x_{37} + 50x_{45} + 132x_{46} + 130x_{58}$$

$$+ 84x_{67} + 135x_{68}$$

$$\max \quad X' = 0.59x_{12} + 0.58x_{13} + 0.6x_{14} + 1.94x_{26}$$

$$+ 0.63x_{37} + 0.58x_{45} + 0.72x_{46} + 1.99x_{58}$$

$$+ 0.72x_{67} + 2.72x_{68}$$

$$\min \quad Y = 450y_1 + 660y_2 + 780y_3 + 90y_4 + 110y_5$$

$$+ 33y_6 + 30y_7 + 34y_8$$

s.t.

$$x_{12} + x_{13} + x_{14} + x_{26} + x_{37} + x_{45} + x_{46} + x_{58} + x_{67}$$

$$+ x_{68} = y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 - 1$$

$$x_{14} \leq y_1$$

$$x_{14} \leq y_4$$

$$x_{26} \leq y_2$$

$$x_{26} \leq y_6$$

$$x_{37} \leq y_3$$

$$x_{37} \leq y_7$$

$$x_{46} \leq y_4$$

$$x_{46} \leq y_6$$

$$x_{58} \leq y_5$$

$$x_{58} \leq y_8$$

$$x_{12} \leq y_1$$

$$x_{12} \leq y_2$$

$$x_{67} \leq y_6$$

$$x_{67} \leq y_7$$

$$x_{13} \leq y_1$$

$$x_{13} \leq y_3$$

$$x_{45} \leq y_5$$

$$x_{45} \leq y_4$$

$$x_{68} \leq y_6$$

$$x_{68} \leq y_8$$

$$x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_2 + y_4$$

$$x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_2 + y_6$$

$$x_{14} + x_{26} + x_{12} + x_{46} \leq y_1 + y_6 + y_4$$

$$x_{14} + x_{26} + x_{12} + x_{46} \leq y_6 + y_2 + y_6$$

$$x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_5 + y_6$$

$$x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_5 + y_8$$

$$x_{45} + x_{58} + x_{68} + x_{46} \leq y_4 + y_8 + y_6$$

$$x_{45} + x_{58} + x_{68} + x_{46} \leq y_8 + y_5 + y_6$$

$$x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_3 + y_6$$

$$x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_3 + y_7$$

$$x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_2 + y_7 + y_6$$

$$x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_1 + y_7 + y_3 + y_6$$

$$x_{12} + x_{26} + x_{67} + x_{37} + x_{13} \leq y_7 + y_2 + y_3 + y_6$$

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

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

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

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

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

$$x_{14} + x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_1 + y_2 + y_4 + y_5 + y_6$$

$$x_{14}+x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_1 + y_2 + y_4 + y_5 + y_8$$

$$x_{14}+x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_1 + y_2 + y_4 + y_8 + y_6$$

$$x_{14}+x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_1 + y_2 + y_8 + y_5 + y_6$$

$$x_{14}+x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_1 + y_8 + y_4 + y_5 + y_6$$

$$x_{14}+x_{45} + x_{58} + x_{26} + x_{12} + x_{68}$$

$$\leq y_8 + y_2 + y_4 + y_5 + y_6$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_3 + y_4 + y_5 + y_6 + y_7$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_3 + y_4 + y_5 + y_6 + y_8$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_3 + y_4 + y_5 + y_8 + y_7$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_3 + y_4 + y_8 + y_6 + y_7$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_3 + y_8 + y_5 + y_6 + y_7$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_1 + y_8 + y_4 + y_5 + y_6 + y_7$$

$$x_{14}+x_{45} + x_{58} + x_{68} + x_{37} + x_{67} + x_{13}$$

$$\leq y_8 + y_3 + y_4 + y_5 + y_6 + y_7$$

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

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

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

Now, we apply Algorithms 4 and 5 for obtaining the Pareto optimal solutions. After performing Algo-

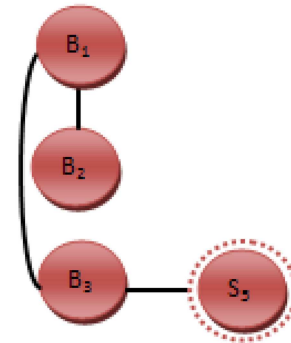


Figure 12. First Pareto optimal solution.

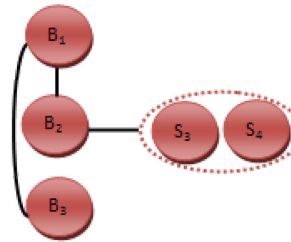


Figure 13. Second Pareto optimal solution.

rithm 4, we obtain the pay-off table as follows:

$$F = \begin{bmatrix} -3515 & -9.17 & 2187 \\ -899 & -9.32 & 2187 \\ -2910 & -1.8 & 1920 \end{bmatrix}.$$

Then, using Algorithm 5, with $q_2 = q_3 = 10$, we solve 121 linear programming problems, and obtain the following Pareto optimal solutions:

1. The first Pareto optimal solution is $Y^* = 1920$, $X^* = 2910$ and $X'^* = 1.8$, with the optimal network as shown in Figure 12.
2. The second Pareto optimal solution is $Y^* = 1923$, $X^* = 3008$ and $X'^* = 3.11$, with the optimal network as shown in Figure 13.
3. The third Pareto optimal solution is $Y^* = 1957$, $X^* = 3143$ and $X'^* = 5.83$, with the optimal network as shown in Figure 14.
4. The fourth Pareto optimal solution is $Y^* = 1987$, $X^* = 3227$ and $X'^* = 6.55$, with the optimal network as shown in Figure 15.
5. The fifth Pareto optimal solution is $Y^* = 2067$, $X^* = 3273$ and $X'^* = 7.82$, with the optimal network as shown in Figure 16.
6. The sixth Pareto optimal solution is $Y^* = 2097$, $X^* = 3357$ and $X'^* = 8.54$, with the optimal network as shown in Figure 17.
7. The final Pareto optimal solution is $Y^* = 2187$, $X^* = 899$ and $X'^* = 9.32$, with the optimal network as shown in Figure 18.

As shown in Figures 12-18, the proposed method provides different products for producers and con-

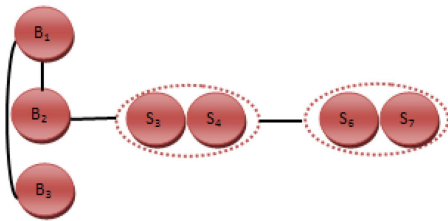


Figure 14. Third Pareto optimal solution.

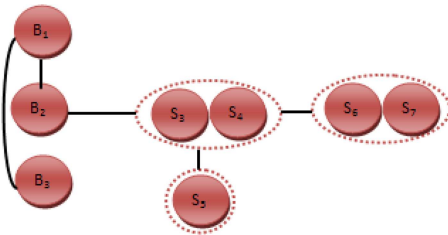


Figure 15. Fourth Pareto optimal solution.

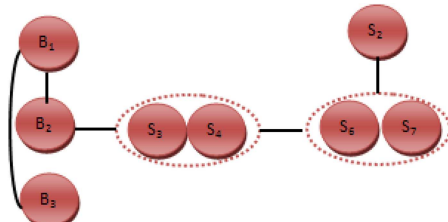


Figure 16. Fifth Pareto optimal solution.

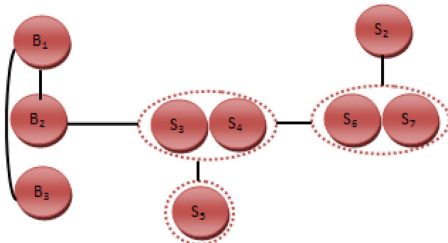


Figure 17. Sixth Pareto optimal solution.

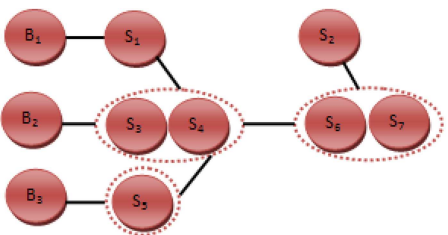


Figure 18. Final Pareto optimal solution.

sumers having different benefits, costs and customer value. For example, consider product 1, which shows a Pareto optimal solution. This product has 3 functions and a subfunction. The production cost for this product is 1920, the benefit is 2910 and customer total value is 1.8. The cost and customer total value of this product is the lowest among others, and the benefit is close to the lowest. On the other hand, product 7 has

the highest customer total value among all products and the lowest benefit. Product 6 has the highest benefit and a high customer total value. The differences in the objectives can help to incorporate customer utilities in a decision making process. The numerical results imply the configuration of different products having various costs and customer values, based on customer views obtained from the web based system. The products themselves are those providing maximum benefits for the producers. The significant decision made in the proposed methodology is the trade-off between cost, benefit, and customer value objectives, which is based on customer views on adding features of products, and producer views on the configuration of beneficial features.

From a managerial perspective, due to rapid changes in products, specifically for digital devices, the need for tactical planning is apparent. Therefore, design of a comprehensive methodology to consider customer opinion in fulfilling those products is valuable. The proposed method is a helpful decision support for managers to make real time decisions with respect to the dynamism in customer views and market pull. This way, customer customization is met, leading to larger market shares and more profit. Managers can determine the demanding subfunctions to fortify the engineering design and production unit for more profit.

If we consider the objectives separately, along with the constraints, the three optimal solutions are characterized as follows:

- If the cost objective only is considered, then, the objective value is $Y^* = 1920$, with the optimal solution as shown in Figure 12.
- If the customer's value objective function is considered, then the optimal objective value is $X'^* = 9.32$, with the Pareto solution as shown in Figure 18.
- If the benefit objective function is used, then the optimal value is $X^* = 3515$, with the optimal solution as shown in Figure 19.

If the number of grid points in the ε -constraint method is increased, then, the optimal solution is expected to change as well. This way, different settings of different objectives can be obtained for convergent products.

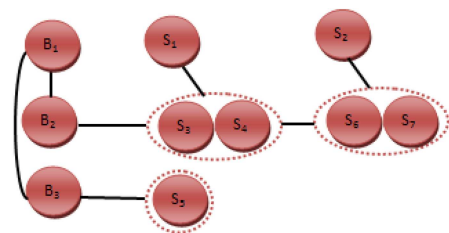


Figure 19. Optimal solution with benefit as objective function.

5. Conclusions

In this paper, we have proposed a methodology to determine value adding functionalities for convergent products. A collection of base functions and sub-functions configure the nodes of a web-based (digital) network representing functionalities. Each arc in the network is to be assigned as the 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, 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 views obtained from a web based system. The products themselves are those that provide maximum benefit for the producers. The important result obtained from the proposed methodology is that customer values, corresponding to benefits, are considered for addition of features to products and producers. An ε -constraint approach was employed to optimize the proposed multi-objective model.

Acknowledgments

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

References

- Yoffie, D.B. "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 (1997).
- Gill, T. "Convergent products: What functionalities add more value to the base?", *Journal of Marketing*, **72**(2), pp. 46-62 (2008).
- Gill, T. and Lei, J. "Convergence in the high-technology consumer markets: Not all brands gain equally from adding new functionalities to products", *Marketing Letters*, **20**(1), pp. 91-103 (2009).
- Callahan, J. and Lasry, E. "The importance of customer input in the development of very new products", *R&D Management*, **34**(2), pp. 107-120 (2004).
- Verworn, B., Herstatt, C. and Nagahira, A. "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), pp. 1-19 (2008).
- Lee, C., Song, B. and Park, Y. "Design of convergent product concepts based on functionality: An association rule mining and decision tree approach", *Expert Systems with Applications*, **39**, pp. 9534-9542 (2012).
- Johnson, D.S. "The NP-completeness column: An ongoing guide", *Journal of Algorithms*, **6**, pp. 434-451 (1985).
- Hakimi, S.B. "Steiner's problem in graphs and its implications", *Networks*, **1**, pp. 113-133 (1971).
- Lawler, E.L., *Combinatorial Optimization Networks and Matroids*, Holt, Rinehart and Winston, New York (1976).
- Hwang, F.K. and Richards, D.S. "Steiner tree problems", *Networks*, **22**, pp. 55-89 (1992).
- Alvarez-Miranda, E., Ljubic, I. and Toth, P. "Exact approaches for solving robust prize-collecting Steiner tree problems", *European Journal of Operational Research*, **229**(3), pp. 599-612 (2013).
- Fu, Z.H. and Hao, J.K. "Breakout local search for the Steiner tree problem with revenue, budget and hop constraints", *European Journal of Operational Research*, **232**, pp. 209-220 (2014).
- Dreyfuss, S.E. and Wagner, R.A. "The Steiner problem in graphs", *Networks*, **1**, pp. 195-207 (1971).
- Beasley, J.E. "An SST-based algorithm for the Steiner problem in graphs", *Networks*, **19**, pp. 1-16 (1989).
- Koch, T. and Martin, A. "Solving Steiner tree problems in graphs to optimality", *Networks*, **32**, pp. 207-232 (1998).
- Duin, C.W. and Volgenant, A. "Reduction tests for the Steiner problem in graphs", *Networks*, **19**, pp. 549-567 (1989).
- Takahashi, H. and Matsuyama, A. "An approximate solution for the Steiner problem in graphs", *Mathematica Japonica*, **24**(6), pp. 573-577 (1980).
- Kou, L., Markowsky, G. and Berman, L. "A fast algorithm for Steiner trees", *Acta Informatica*, **15**, pp. 141-145 (1981).
- Rayward-Smith, V.J. and Clare, A. "On finding Steiner vertices", *Networks*, **16**, pp. 283-294 (1986).
- Winter, P. and MacGregor Smith, J. "Path-distance heuristics for the Steiner problem in undirected networks", *Algorithmica*, **7**, pp. 309-327 (1992).
- Mehlhorn, K. "A faster approximation algorithm for the Steiner problem in graphs", *Information Processing Letters Archive*, **27**, pp. 125-128 (1988).
- Robins, G. and Zelikovsky, A. "Improved Steiner tree approximation in graphs", in *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Al-*

- gorithms*, SIAM, Philadelphia, ACM, New York, pp. 770-779 (2000).
23. Robins, G. and Zelikovsky, A. "Tighter bounds for graph Steiner tree approximation", *SIAM Journal on Discrete Mathematics*, **19**(1), pp. 122-134 (2005).
 24. Esbensen, H. "Computing near-optimal solutions to the Steiner problem in a graph using a genetic algorithm", *Networks*, **26**, pp. 173-185 (1995).
 25. Kapsalis, A., Rayward-Smith, V.J. and Smith, G.D. "Solving the graphical Steiner tree problem using genetic algorithms", *Journal of the Operational Research Society*, **44**(4), pp. 397-406 (1993).
 26. Martins, S.L., Pardalos, P., Resende, M.G. and Ribeiro, C.C. "Greedy randomized adaptive search procedures for the Steiner problem in graphs", *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, **43**, pp. 133-146 (1999).
 27. Ribeiro, C.C. and Souza, M.C. "Tabu search for the Steiner problem in graphs", *Networks*, **36**, pp. 138-146 (2000).
 28. Martins, S.L., Ribeiro, C.C. and Souza, M.C. "A parallel GRASP for the Steiner problem in graphs", *Lecture Notes in Computer Science*, Springer-Verlag, **1457**, pp. 310-331 (1998).
 29. Martins, S.L., Resende, M.G.C., Ribeiro, C.C. and Pardalos, P.M. "A parallel GRASP for the Steiner tree problem in graphs using a hybrid local search strategy", *Journal of Global Optimization*, **17**, pp. 267-283 (2000).
 30. Di Fatta, G., Lo Presti, G. and Lo Re, G. "A parallel genetic algorithm for the Steiner problem in networks", in *Proceedings of the 15th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2003)*, Marina del Rey, CA, USA, pp. 569-573 (2003).
 31. Costa, A.M., Cordeau, J.F. and Laporte, G. "Steiner tree problems with profits", *INFOR*, **44**(2), pp. 99-115 (2006).
 32. Chankong, V. and Haimes, Y.Y., *Multiobjective Decision Making: Theory and Methodology*, North-Holland, (1983).
 33. Ehrgott, M. and Wiecek, M.M. "Multiobjective programming", in *Multiple Criteria Decision Analysis: State of the Art Surveys*, J. Figueira, S. Greco and M. Ehrgott (Eds.), Kluwer Academic Publishers, pp. 667-722 (2005).
 34. Miettinen, K.M., *Nonlinear Multiobjective Optimization*, Kluwer Academic (1999).
 35. Steuer, R.E., Gardiner, L.R. and Gray, J. "A bibliographic survey of the activities and international nature of multiple criteria decision making", *Journal of Multi-Criteria Decision Analysis*, **5**, pp. 195-217 (1996).
 36. Ehrgott, M. and Gandibleux, X. "Multiobjective combinatorial optimization-theory, methodology, and applications", in *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, M. Ehrgott and X. Gandibleux (Eds.), Kluwer Academic Publishers, pp. 369-444 (2002).
 37. Ulungu, E.L. and Teghem, J. "The two-phase method: An efficient procedure to solve bi-objective combinatorial optimization problems", *Foundations of Computing and Decision Sciences*, **20**, pp. 149-165 (1995).
 38. Kouvelis, P. and Sayin, S. "Algorithm robust for the bicriteria discrete optimization problem: Heuristic variations and computational evidence", *Annals of Operations Research*, **147**, pp. 71-85 (2006).
 39. Ralphs, T.K., Saltzman, M.J. and Wiecek, M.M. "An improved algorithm for solving biobjective integer programs", *Annals of Operations Research*, **147**, pp. 43-70 (2006).
 40. Sayin, S. and Kouvelis, P. "The multiobjective discrete optimization problem: A weighted min-max two-stage optimization approach and a bicriteria algorithm", *Management Science*, **51**, pp. 1572-15819 (2005).
 41. Cohon, J.L., *Multiobjective Programming and Planning*, New York, Academic Press (1978).
 42. Wood, A.J. and Wollenberg, B.F., *Power Generation, Operation and Control*, 2nd ed., New York, Wiley (1996).
 43. Mavrotas, G. "Effective implementation of the e-constraint method in multiobjective mathematical programming problems", *Appl. Math. Comput.*, **213**(2), pp. 455-65 (2009).
 44. Aghaeia, J., Amjady, N. and Shayanfar, H.A. "Multi-objective electricity market clearing considering dynamic security by lexicographic optimization and augmented epsilon constraint method", *Applied Soft Computing*, **11**, pp. 3846-3858 (2011).
 45. Kazemi, A., Rezaei-pour, R. and Lashkarara, A. "Optimal location of Rotary Hybrid Flow Controller (RHFC) through multi-objective mathematical programming", *Scientia Iranica*, **19**(6), pp. 1771-1779 (2012).
 46. Lashkar Ara, A., Kazemi, A., Gahramani, S. and Behshad, M. "Optimal reactive power flow using multi-objective mathematical programming", *Scientia Iranica*, **19**(6), pp. 1829-1836 (2012).

Biographies

Reza Hassanzadeh is a PhD degree student of Industrial Engineering at Mazandaran University of Science and Technology, in Iran. His research interests include optimization, mathematical programming, network theory, and meta-heuristic approaches. He has published over 35 papers in international journals and conferences.

Iraj Mahdavi received his PhD degree from India in Production Engineering, and his Post-Doctorate degree from Hanyang University, Korea. He is currently Professor of Industrial Engineering and Vice President of Graduate Studies and Research at Mazandaran University of Science and Technology, Iran. He is also on the Editorial Board of four journals, and has published over 250 research papers. His research interests include cellular manufacturing, digital management of industrial enterprises, intelligent operations management, and industrial strategy settings.

Nezam Mahdavi-Amiri received his PhD degree from Johns Hopkins University, USA, in Mathematical Sciences, and is currently Professor of Mathematical Sciences at Sharif University of Technology, Tehran, Iran. He is on the Editorial Board of several international mathematical and computing journals, including the Bulletin of the Iranian Mathematical Society, The CSI Journal of Computer Science and Engineering, and the Journal of the Iranian Operations Research Society. His research interests include optimization, numerical analysis, and scientific computing, matrix computations, fuzzy modeling and computing.