A metaheuristic algorithm to solve the selection of transportation channels in supply chain design

Elias Olivares-Benitez[a], Roger Z. Ríos-Mercado[b], José Luis González-Velarde[c]

[a] UPAEP University, 21 Sur 1103, Puebla, Puebla, 72410, Mexico, elias.olivares@upaep.mx, Tel. +52-222-229-9400 x 7783, Corresponding autor

[b] Graduate Program in Systems Engineering, Universidad Autónoma de Nuevo León, AP 111-F, Cd. Universitaria, San Nicolás de los Garza, Nuevo León, 66450, Mexico, roger.rios@uanl.edu.mx

[c] Tecnologico de Monterrey, E. Garza Sada Sur 2501, Monterrey, Nuevo León, 64849, Mexico, gonzalez.velarde@itesm.mx

Abstract

This paper addresses a supply chain design problem based on a two-echelon single-product system. In the first echelon the plants transport the product to distribution centers. In the second echelon the distribution centers transport the product to the customers. Several transportation channels are available between nodes in each echelon, with different transportation costs and times. The decision variables are the opening of distribution centers from a discrete set, the selection of the transportation channels, and the flow between facilities. The problem is modeled as a bi-objective mixed-integer program. The cost objective aggregates the opening costs and the transportation costs. The time objective considers the longest transportation time from the plants to the customers. An implementation of the classic epsilon-constraint method was used to generate true efficient sets for small instances of the problem, and approximate efficient sets for larger instances. A metaheuristic algorithm was developed to solve the problem, as the major contribution of this work. The metaheuristic algorithm combines principles of greedy functions, Scatter Search, Path Relinking and Mathematical Programming. The large instances were solved with the metaheuristic algorithm and a comparison was made in time and quality with the epsilon-constraint based algorithm. The results were favorable to the metaheuristic algorithm for large instances of the problem.

Keywords: metaheuristic; multiobjective; supply chain design; location; transportation.

1. Introduction

In recent years Supply Chain Design has been addressed by many authors, and several reviews have been published (Aikens, 1985; Thomas and Griffin, 1996; Vidal and Goetschalckx, 1997; Beamon, 1998; Klose and Drexel, 2005; Sahin and Sural, 2007; Melo et al., 2009). The decisions imply strategic aspects related with location, capacities and technology selection, and tactical aspects like product allocation and transportation flows, among others.

In this paper we address a previous work by the authors (Olivares-Benitez et al., 2012) where a supply chain design problem, based on a two-echelon single-product system was introduced. The problem considers the location of facilities, the selection of transportation channels, the calculation of the flows between facilities, and the time-cost tradeoff. In particular, the selection of transportation channels produces a bi-objective optimization problem where cost and lead time must be minimized. The transportation channels can be seen as transportation modes (rail, truck, ship, airplane, etc.), shipping services (express, normal, overnight, etc.) or as transportations offers from different companies. Each option has a cost and time associated, and one must be selected to transport the product between nodes in each echelon. The problem was solved in an *a posteriori* approach, obtaining the non-dominated solutions set to be presented to the decision maker.

The objective in this new research was to develop a metaheuristic algorithm to solve the problem introduced by Olivares-Benitez et al. (2012). It was demonstrated that the problem belongs to the NP-Hard type. Hence it is necessary to use a heuristic method to solve large instances of the problem. The metaheuristic algorithm proposed here hybridizes elements from greedy functions, Scatter Search, Path Relinking, and Mathematical Programming. This type of hybrids, also named matheuristics, is being used in recent research but there are not applications in supply chain design yet.

The review in Section 2 describes works that connect the cost-time tradeoff in supply chain design, and in the most recent studies, the consideration of time tied to transportation decisions in multiobjective problems. According to the analysis, the use of matheuristic algorithms and transportation channel selection in the context of supply chain design represent major contributions of this paper.

The problem addressed along with the mathematical model is described in detail in Section 3. The methods used to solve the problem are detailed in Section 4. For small instances the epsilon-constraint based algorithm proposed by Olivares-Benitez et al. (2012) was used to obtain the true efficient sets. The largest instance solved with the epsilon-constraint based algorithm to obtain its true efficient set has 5 plants, 5 potential distribution centers, and 20 customers. To construct approximate efficient sets for larger instances the same method was used with a time limit of 3600 seconds per point. Given the complexity of the problem, a metaheuristic algorithm was developed in this work to obtain approximate efficient sets for large instances. The largest instance where an approximate efficient set was obtained has 50 plants, 50 potential distribution centers, and 100 customers. The generation of instances and the computational evaluation are described in Section 5. Finally, Section 6 presents the conclusions of this work.

2. Literature Review

One characteristic that differentiates the problem introduced by Olivares-Benitez et al. (2012) from previous works in the literature is the study of the tradeoff between lead time and cost in the supply chain design, related to transportation choices. The review by Current et al. (1990) makes evident that the balance of these criteria had not been studied extensively. After that, Arntzen et al. (1995) addressed the supply chain design problem for a company that handled the cost-time tradeoff as a weighted combination in the objective function. The decision variable was the quantity of product to be sent through each transportation mode available. Transportation time was variable with respect to the quantity shipped. The problem was solved using elastic penalties for violating constraints, and a row-factorization technique. Zeng (1998) emphasized the importance of the lead time-cost tradeoff, associated to the transportation modes available between pairs of nodes in the network. A mixed-integer programming model was proposed to design the supply chain optimizing both objectives. In this work facility location was not addressed. The method proposed was a dynamic programming algorithm to construct the efficient frontier assuming the discretization of time. In the model proposed by Graves and Willems (2005) cost and time were combined in the objective function. The supply chain was configured selecting alternatives at each stage of the production and distribution network. A dynamic programming algorithm was used to solve this problem.

In recent years multiobjective problems in supply chain design have been treated with more emphasis taking advantage of increased computational resources and new methods. Chan et al. (2006) presented a multi-objective model that optimized a combined objective function with weights. Some of the criteria included cost and time functions, and one of the components of time was transportation time. Transportation time varied linearly with the quantity transported. The model included stochastic components, but facility location was not considered. A genetic algorithm was the base of an iterative method where scenarios with changing weights were solved. Altiparmak et al. (2006) proposed a model with three objective functions: to minimize total cost, to maximize total customer demand satisfied, and to minimize the unused capacity of distribution centers. Here, transportation time was handled as a constraint that determined a set of feasible distribution centers able to deliver the product to the customer before a due date. They proposed a procedure based on a genetic algorithm to obtain a set of non-dominated solutions. In the work by ElMaraghy and Majety (2008) a model was proposed to optimize cost, including the cost of late delivery. The model considered the dynamic nature of the decisions. They used commercial optimization software to solve the model, analyzing different scenarios. The review by Farahani et al. (2010) about multi-criteria models for facility location problems describes some works where metrics of cost and service level are considered. The metaheuristic methods mentioned include multiobjective versions of Scatter Search, Tabu Search, Simulated Annealing, Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO). However, some other metaheuristics that were created for multiobjective applications were also mentioned, like Simple Evolutionary Algorithm for Multi-Objective Optimization (SEAMO), Strength Pareto Evolutionary Algorithm version 2 (SPEA2), Pareto Envelop based Selection Algorithm (PESA), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Vector Evaluated Genetic Algorithm (VEGA), and the Multi-Objective Genetic Algorithm (MOGA).

More recently, several works have appeared for multiobjective supply chain design. Pishvaee et al. (2010) studied a model for a forward/reverse logistics network design from a bi-objective optimization perspective. The objectives to optimize were the total cost of the system and the fulfillment of the demand and return rates. Although they considered lead time into their model, similar to Altiparmak et al. (2006) it was a considered in the meeting of a due date, and not related to transportation alternatives. They developed a memetic algorithm to solve this NP-hard problem. Moncayo-Martinez and Zhang (2011) proposed a model similar to that of Graves and Willems (2005) where activities must be selected to design the supply chain. This was a bi-objective model that optimized cost and lead time in a multi-echelon network. The decision variable is the selection of the resource for a certain activity in the supply chain. They used a Pareto Ant Colony Optimization metaheuristic to obtain the Pareto Optimal Set. Liao et al. (2011) also studied a multiobjective problem for supply chain design. In this case they integrated location and inventory decisions. The objectives were the minimization of cost, the maximization of the fill rate, and the maximization of demand fulfilled within a coverage distance. The lead time was implied in the cost of the safety stock, but it was not related to transportation decisions. The method proposed was a hybrid of NSGA-II and an assignment heuristic. Pinto-Varela et al. (2011) presented a bi-objective optimization model for the design of supply chains considering economic and environmental criteria. In their model, time was considered since the point of view of a multi-period approach. Different transportation modes may exist, but they are not associated to the time.

They solved three small examples with mathematical programming commercial software. The review by Mansouri et al. (2012) emphasized the importance of multiobjective optimization techniques as decision support tool in supply chain management. Although order promising decisions and network design decisions were identified as important criteria, none of the works reviewed integrated them in a multiobjective approach. Chaabane et al. (2012) presented a multi-period multiobjective optimization problem where cost and environmental objectives were optimized. In their mixed-integer programming model, the selection of transportation modes was considered as a decision variable but it was not connected with time. They used mathematical programming commercial software to solve small instances of the problem. Sadjady and Davoudpour (2012) studied a problem for supply chain design where cost and time were tied to transportation alternatives. The approach, however, was to optimize a single objective function where lead time from the transportation alternative was transformed into a cost function. The cost objective function is optimized using a Lagrangian relaxation method. As proposed by Olivares-Benitez et al. (2012), the cost and time criteria may not be comparable and should be treated in separate objectives.

It is important to highlight some works that solve real cases for supply chain design. Altiparmak et al. (2006) applied their genetic algorithm for a supply chain design for plastic products in Turkey. Pati et al. (2008) solved a case for the Indian paper recycling industry. Sousa et al. (2008) applied their models for the design of an agrochemicals supply chain. Gumus et al. (2009) solved the case for a company in the alcohol free beverage sector. Moncayo-Martinez and Zhang (2011) applied a Pareto Ant Colony Optimization metaheuristic to design a supply chain for Bulldozer production. Pinto-Varela et al. (2011) presented a bi-objective model for designing supply chains in Portugal. Chaabane et al. (2012) solved a case for aluminum production. Funaki (2012) proposed a very complete model and a dynamic programming algorithm to design a supply chain for a machinery product. Marvin et al. (2012) formulated a mixed integer linear programming problem to design a supply chain for ethanol biorefining. Paksoy et al. (2012) applied fuzzy optimization for the design of a vegetable oil supply chain. These works illustrate an increasing interest in the application of supply chain design models in industry.

Finally, it is interesting to note the review by Griffis et al. (2012) where they presented the use of metaheuristics in logistics and supply chain management from year 1991 to 2012. Near 15% of the applications were in the area of supply chain design. They highlight the use of Simulated Annealing and Tabu Search among local search metaheuristics, with minor attention in the literature to greedy randomized adaptive search procedure (GRASP), variable neighborhood search (VNS) and others. In terms of population search techniques, the most popular have been Genetic Algorithms and Ant Colony Optimization, with fewer mentions for Scatter Search, Particle Swarm Optimization, and others. However in this review it is evident the few applications of multiobjective metaheuristics, especially for supply chain design problems.

The research described above shows that few works considered the cost–time tradeoff derived from the transportation channel selection in the supply chain design. Other differences with the problem addressed in this research are explained in the following lines. First, in some works the transportation time is a linear function of the quantity transported. In the model presented here, a single time is used for each arc between nodes, which represents more real conditions in the operation of transportation. Second, in many studies the time-cost tradeoff has been addressed from a single objective perspective transforming the time in a cost function. Here, the time and cost are treated as separate criteria allowing for the construction of sets of non-dominated solutions. This approach may be a good choice when the preference of the decision maker for one of the objectives is not known, or when the criteria cannot be compared easily. Third, in many multiobjective problems for supply chain design, the cost-time tradeoff was not associated to the selection of the transportation channel. In the problem addressed here, the selection of transportation from several alternatives has a direct impact in the lead time objective. The combination of these elements and traditional supply chain design decisions makes relevant the problem addressed, and the necessity to solve it.

In terms of the algorithm developed here, what we propose is a hybridization of greedy functions with Scatter Search, Path Relinking and mathematical programming software, which produces high quality solutions for a complex problem. In the literature, the techniques preferred to solve these multiobjective problems with the *a posteriori* approach are variations of evolutionary algorithms. The type of hybrids presented in this work, also named matheuristics, has not been used before in the context of supply chain design problems. However other applications can be seen in the book edited by Maniezzo et al. (2010).

3. Problem description and mathematical model

The problem introduced by Olivares-Benítez et al. (2012) was a two-echelon distribution system for one product in a single time period. A set of manufacturing plants produce and send the product to distribution centers in the first stage. Later, the distribution centers transport the product to the customers. The number and location of plants and customers, along with demands and capacities respectively, are known. The distribution centers must be selected from a discrete set of potential locations with fixed opening costs and limited capacities. A single sourcing policy was assumed for the transportation from the distribution centers to the customers. Figure 1 depicts the structure of the supply chain.

[Figure 1 goes about here]

The transportation of the product from one facility to the other in each echelon of the network is done selecting one of several alternatives available. Each transportation channel represents a type of service with associated cost and time parameters. These alternatives can be obtained from offers of different companies, the availability of different types of service for each company (e.g. express and regular), or the use of different modes of transportation (e.g. truck, rail, airplane, ship or inter-modal). It was assumed that a faster service is usually more expensive. The capacity of the transportation channel was assumed as unlimited, considering that any capacity can be contracted.

A bi-objective mixed-integer programming model was proposed to solve the problem described previously, as follows.

Sets:

I : set of plants i
J : set of potential distribution centers j
K : set of customers k
LP_{ij} : set of arcs l between nodes i and j; $i \in I, j \in J$
LW_{jk} : set of arcs l between nodes j and k; $j \in J, k \in K$

Parameters:
CP_{ijl} : cost of transporting one unit of product from plant i to distribution center j using arc l; $i \in I, j \in J, l \in LP_{ij}$
CW_{jkl} : cost of sending one unit of product from distribution center j to customer k using arc l; $j \in J, k \in K, l \in LW_{jk}$
TP_{ijl} : time for transporting any quantity of product from plant i to distribution center j using arc l; $i \in I, j \in J, l \in LP_{ij}$
TW_{jkl} : time for transporting any quantity of product from distribution center j to customer k using arc l; $j \in J, k \in K, l \in LW_{jk}$
MP_i : capacity of plant i; $i \in I$
MW_j : capacity of distribution center j; $j \in J$
D_k : demand of customer k; $k \in K$
F_j : fixed cost for opening distribution center j; $j \in J$

Decision variables:
X_{ijl} : quantity transported from plant i to distribution center j using arc l; $i \in I, j \in J, l \in LP_{ij}$
Y_{jkl} : quantity transported from distribution center j to customer k using arc l; $j \in J$, $k \in K, l \in LW_{jk}$
Z_j : binary variable equal to 1 if distribution center j is open and equal to 0 otherwise; $j \in J$
A_{ijl} : binary variable equal to 1 if arc l is used to transport product from plant i to distribution center j and equal to 0 otherwise; $i \in I, j \in J, l \in LP_{ij}$
B_{jkl} : binary variable equal to 1 if arc l is used to transport product from distribution center j to customer k and equal to 0 otherwise; $j \in J, k \in K, l \in LW_{jk}$

Auxiliary variables:
T : longest time that takes sending product from any plant to any customer
E_j^1 : longest time in the first echelon of the supply chain for active distribution center j, i.e. $E_j^1 = \max_{i,l}\left(TP_{ijl} A_{ijl}\right)$; $i \in I, j \in J, l \in LP_{ij}$
E_j^2 : longest time in the second echelon of the supply chain for active distribution center j, i.e. $E_j^2 = \max_{k,l}\left(TW_{jkl} B_{jkl}\right)$; $j \in J, k \in K, l \in LW_{jk}$

MODEL 1:

$$\min\left(f_1, f_2\right)$$

$$f_1 = \sum_{i\in I}\sum_{j\in J}\sum_{l\in LP_{ij}} CP_{ijl}X_{ijl} + \sum_{j\in J}\sum_{k\in K}\sum_{l\in LW_{jk}} CW_{jkl}Y_{jkl} + \sum_{j\in J} F_j Z_j \tag{1}$$

$$f_2 = T \tag{2}$$

subject to

$$T - E_j^1 - E_j^2 \geq 0 \qquad j\in J \tag{3}$$

$$E_j^1 - TP_{ijl}A_{ijl} \geq 0 \qquad i\in I,\ j\in J, l\in LP_{ij} \tag{4}$$

$$E_j^2 - TW_{jkl}B_{jkl} \geq 0 \qquad j\in J,\ k\in K, l\in LW_{jk} \tag{5}$$

$$\sum_{j\in J}\sum_{l\in LW_{jk}} Y_{jkl} = D_k \qquad k\in K \tag{6}$$

$$\sum_{j\in J}\sum_{l\in LP_{ij}} X_{ijl} \leq MP_i \qquad i\in I \tag{7}$$

$$MW_j Z_j - \sum_{k\in K}\sum_{l\in LW_{jk}} Y_{jkl} \geq 0 \qquad j\in J \tag{8}$$

$$\sum_{i\in I}\sum_{l\in LP_{ij}} X_{ijl} - \sum_{k\in K}\sum_{l\in LW_{jk}} Y_{jkl} = 0 \qquad j\in J \tag{9}$$

$$\sum_{j\in J}\sum_{l\in LW_{jk}} B_{jkl} = 1 \qquad k\in K \tag{10}$$

$$\sum_{l\in LP_{ij}} A_{ijl} \leq 1 \qquad i\in I,\ j\in J \tag{11}$$

$$\sum_{l\in LW_{jk}} B_{jkl} \leq 1 \qquad j\in J, k\in K \tag{12}$$

$$X_{ijl} - A_{ijl} \geq 0 \qquad i\in I,\ j\in J, l\in LP_{ij} \tag{13}$$

$$Y_{jkl} - B_{jkl} \geq 0 \qquad j\in J,\ k\in K, l\in LW_{jk} \tag{14}$$

$$MP_i A_{ijl} - X_{ijl} \geq 0 \qquad i\in I,\ j\in J, l\in LP_{ij} \tag{15}$$

$$MW_j B_{jkl} - Y_{jkl} \geq 0 \qquad j\in J,\ k\in K, l\in LW_{jk} \tag{16}$$

$$\sum_{i\in I}\sum_{l\in LP_{ij}} A_{ijl} - Z_j \geq 0 \qquad j\in J \tag{17}$$

$$T, E_j^1, E_j^2, X_{ijl}, Y_{jkl} \geq 0 \qquad i\in I,\ j\in J, k\in K, l\in LP_{ij}, l\in LW_{jk} \tag{18}$$

$$Z_j, A_{ijl}, B_{jkl} \in \{0,1\} \qquad i\in I,\ j\in J, k\in K, l\in LP_{ij}, l\in LW_{jk} \tag{19}$$

In this model, objective function (1) minimizes the sum of the transportation cost and the cost for opening distribution centers. Objective function (2) minimizes the longest transportation time from the plants to the customers through each distribution center. Constraints (3)-(5) calculate the longest transportation time in each echelon for each distribution center. Constraints (6) force the demand satisfaction for each customer. Constraints (7) imply that the capacities of the plants are not exceeded. Constraints (8) meet two conditions: that the flow going out from a distribution center must not exceed its capacity, and that the flow of product is done only through open distribution centers. Constraints (9) keep the flow balance at each distribution center. Constraints (10) force the single source policy from distribution centers to customers. The selection of only one transportation channel between facilities is required in constraints (11) and (12). Constraints (13)-(17) establish links between the sets of variables A_{ijl}, B_{jkl}, X_{ijl}, Y_{jkl} and Z_j to avoid incoherent solutions. Constraints (18) and (19) are for declaration of variables.

About the computational complexity of the problem, it has been demonstrated that the well-known UFLP (Uncapacitated Fixed-Charge Facility Location Problem) is polynimially reducible to the model described above (Olivares-Benitez et al., 2012). Since the UFLP is NP-hard (Cornuejols et al., 1990), the model above is NP-hard too.

4. Exact and metaheuristic methods

4.1 Exact method

The method selected for generating true efficient sets was the epsilon-constraint method. In a multiobjective optimization problem, this method optimizes a series of single objective sub-problems. One of the objective functions is selected to be optimized and the other objective functions are transformed into constraints and added to the set of constraints, as follows.

$$\min\{f_k(x) : f_i(x) \leq \varepsilon_i, i \neq k, x \in X\}$$

Where $f = (f_1,\dots,f_p)$ is the set of p real-valued objective functions, x is a solution to the problem and X is the set of feasible solutions. The values of vectors ε_i are changed systematically to obtain the efficient frontier for the problem. Further details can be seen in Steuer (1989) and Ehrgott (2005) as references.

Olivares-Benitez et al. (2012) developed an implementation of the epsilon-constraint method that uses the solutions generated during the process to accelerate the construction of the true efficient set. This version of the epsilon-constraint method, named "Backward epsilon-constraint method with estimated lower limit for f_2" (ReC), was used to construct the true efficient sets for several small instances generated artificially. The procedure was coded in ANSI C. The single-objective subproblems of the epsilon-constraint based algorithm were solved using the CPLEX 11.1 callable library (ILOG, 2008).

4.2 Metaheuristic method

Because of the computational complexity of the problem, relatively large instances may no longer be tractable from an exact optimization perspective. Thus the development of a heuristic method is suitable to find an approximate set of efficient solutions. In this work we propose a metaheuristic algorithm to approximate efficient sets of the problem for large instances. This is a population-based metaheuristic that uses some principles of Scatter Search, Path Relinking (Laguna and Marti, 2003), greedy functions and mathematical programming. Historically, greedy functions have been used in the design of heuristics to solve hard combinatorial optimization problems. In the framework of metaheuristics, greedy functions are used to construct initial solutions in the GRASP (Resende and Ribeiro, 2003;

Talbi, 2009) and are used for a good approximation in other local search heuristics. Several metaheuristic implementations have combined effectively GRASP and Scatter Search or GRASP and Path Relinking to tackle difficult combinatorial optimization problems, and recently this hybridization has been proposed for multiobjective combinatorial optimization problems (Marti et al., 2011). In the metaheuristic proposed in this work we wanted to keep the good approximation achieved by greedy functions with the population-based approach given by Scatter Search to construct approximate efficient sets. Additionally, the use of Path Relinking was added to improve the quality of the solutions obtained in a final combination stage. This hybrid also uses mathematical programming software embedded into the metaheuristic algorithm. This idea is being studied in recent works (Maniezzo et al. 2010) to increase the power of metaheuristic algorithms.

The metaheuristic algorithm proposed in this work is composed of three main methods. These are a constructive method, an improvement method, and a combination method. These methods use a basic procedure to construct a solution based on a decomposition of the problem. It is important to explain this hierarchical construction procedure before going to the details of the methods.

4.2.1 Hierarchical construction procedure

A solution is constructed hierarchically starting with the selection of the distribution centers to be opened. Each method uses a specific strategy to perform this selection as will be described below. The next decision in the hierarchy is the selection of the transportation channel between each pair of facilities. The selection of the transportation channel is done using a weighted greedy function. This greedy function has a component based on the transportation cost and the other component based on the transportation time as shown in equations (20) and (21). These functions are normalized to avoid the scaling problem. A higher value of the greedy function implies a worse selection considering that both criteria, time and cost, are minimized:

$$\phi\left(arc_{ijl}\right) = \lambda_c \frac{CP_{ijl}}{\max\limits_{i \in I, j \in J, l \in LP_{ij}}\left(CP_{ijl}\right)} + \lambda_t \frac{TP_{ijl}}{\max\limits_{i \in I, j \in J, l \in LP_{ij}}\left(TP_{ijl}\right)} \qquad (20)$$

$$\phi\left(arc_{jkl}\right) = \lambda_c \frac{CW_{jkl}}{\max\limits_{j \in J, k \in K, l \in LW_{jk}}\left(CW_{jkl}\right)} + \lambda_t \frac{TW_{ijl}}{\max\limits_{j \in J, k \in K, l \in LW_{jk}}\left(TW_{jkl}\right)} \qquad (21)$$

The weights λ_c and λ_t for the greedy functions are systematically changed each iteration of the constructive method and inherited through the rest of the algorithm. The aim of weights variation is to obtain solutions well distributed along the efficient frontier instead of a concentration of solutions in the extremes of the frontier. More details about the procedure to calculate these weights are given below in the explanation of the constructive method.

Once the transportation channel with the best value is selected, the problem can be decomposed by echelon. First, the flow of product from distribution centers to the customers can be obtained solving a generalized assignment problem (GAP) as depicted in Figure 2. The solution to the GAP assigns customers to distribution centers, and all the demand of the customer is satisfied by the distribution center assigned. The costs used in the formulation of the GAP correspond to the values of the greedy functions $\phi(arc_{jkl})$. Later, the flow of product from the plants to the distribution centers is obtained solving a transportation problem (TP) as shown in Figure 2. The demand at the open distribution centers is the sum of the demands of the customers assigned to them previously. In this step the costs in the TP are the values of the greedy functions $\phi(arc_{ijl})$. This basic procedure is called to construct a solution in each method. The GAP and the TP are solved using mathematical programming commercial software.

[Figure 2 goes about here]

4.2.2 General algorithm

The scheme of the algorithm proposed is presented in Figure 3. A strategy of elitism is used to avoid losing solutions after each method and then converging toward the true efficient set. The solutions from the constructive and improvement methods are used to update the approximate efficient set *NDS* (Non-dominated Solutions) using the dominance relation of the new solutions with respect to those already in *NDS*. After the execution of each method a reference set (*RS*) is constructed combining the solutions in the updated set *NDS* and the "diverse" solutions obtained from the method. The diverse solutions are selected among those close to the current set *NDS* in the objective functions space. The dotted lines in Figure 4 represent calls and updates to the *NDS* set. Finally, in the post-processing stage the last set *RS* is used in the combination method. The solutions obtained in this method are used to update the approximate efficient set *NDS*. The final result of the algorithm is the approximate efficient set in the last *NDS* set.

[Figure 3 goes about here]

The constructive method generates a fixed number of solutions. For each solution, the selection of the distribution centers to be opened is done randomly. The weights (λ_c, λ_t) for the greedy functions in equations (20) and (21) are generated systematically for each solution in a linear combination between (1-λ_f, 0) and (0, 1-λ_f), considering the total number of solutions to be generated. These weights λ_c and λ_t are used to select the transportation channels in each solution, and their values are inherited through the rest of the algorithm. Although the distributions centers are selected randomly in the constructive method, they conserve assigned values of λ_c and λ_t for their application in the rest of the algorithm. The parameter λ_f represents the relative frequency of selection of a certain arc or distribution center with respect to the total number of constructed solutions along the iterations. The parameter λ_f is updated each iteration of the constructive method. This long term memory promotes the selection of new elements each iteration of the constructive method. Once the distribution centers are selected for each generated solution, the hierarchical construction procedure is called to complete the construction of that solution. The algorithm for the constructive method is shown in Figure 4.

[Figure 4 goes about here]

The solutions obtained in the constructive method create and update a set of non-dominated solutions called *NDS*. The solutions in *NDS* are included in a reference set named *RS*. To provide variety to the reference set some dominated solutions are included. These dominated solutions are taken from the points closest to the current efficient frontier in *NDS*.

To guide movements in the improvement and combination methods, a greedy function for the distributions centers was formulated, similar to that of the arcs, as shown in equations (22 - 24).

$$\phi^c(dc_j) = \frac{F_j + \sum_{i \in I} MP_i \max_{l \in LP_{ij}}(CP_{ijl}) + \sum_{k \in K} D_k \max_{l \in LW_{jk}}(CW_{jkl}) / MW_j}{\max_{j \in J}\left(F_j + \sum_{i \in I} MP_i \max_{l \in LP_{ij}}(CP_{ijl}) + \sum_{k \in K} D_k \max_{l \in LW_{jk}}(CW_{jkl}) / MW_j\right)} \quad (22)$$

$$\phi^t(dc_j) = \frac{\min_{i \in I, l \in LP_{ijl}}(TP_{ijl}) + \min_{k \in K, l \in LW_{jkl}}(TW_{jkl})}{\max_{j \in J}\left(\min_{i \in I, l \in LP_{ijl}}(TP_{ijl}) + \min_{k \in K, l \in LW_{jkl}}(TW_{jkl})\right)} \quad (23)$$

$$\phi(dc_j) = \lambda_c \phi^c(dc_j) + \lambda_t \phi^t(dc_j) \quad (24)$$

The improvement method uses local search and explores three types of neighborhoods for each solution in the reference set. These correspond to movements of opening, closing and exchange of distribution centers. For each neighborhood a sorted list is created according to the value of the aggregated greedy function $\phi(dc_j)$ in equation (24). Each element in the list is taken at a time in that order as described below:

- Closing of facilities *CN* (*s*). The open distribution centers are sorted in descending order by $\phi(dc_j)$ value, i.e. from worst to best.
- Opening of facilities *ON* (*s*). The closed distribution centers are sorted in ascending order by $\phi(dc_j)$ value, i.e. from best to worst.
- Exchange of facilities *EN* (*s*). The previous two lists are created. One open facility is closed and one closed facility is opened. The lists are explored taking as pivot the list for opening.

To accept one movement the dominance of the new solution is considered. If an infeasible or dominated solution is created by the movement, it is rejected. Figure 5 shows the acceptance criterion and direction of improvement where weakly and strongly non-dominated solutions are accepted. A short term memory is conserved to avoid cycling during the improvement method. After a movement is done, the dashed areas in Figure 5 indicate the direction of improvement allowed for a new movement. The algorithm for the improvement method is shown in Figure 6.

[Figure 5 goes about here]

[Figure 6 goes about here]

After a number of iterations applying the constructive and improvement methods, the combination method is used as a post-processing stage. It is based on Path Relinking (Laguna and Marti, 2003) to obtain a set of solutions for each pair of solutions from a reference set *RS*. One of the solutions is selected as "initiating solution" and the other is selected as "guiding solution". The combination makes movements in the vector of values Z_j of the distribution centers and completes a solution calling to the hierarchical construction procedure. The path is constructed giving priority to closing movements until infeasibility is found. Then, a distribution center that was closed with respect to the guiding solution is now opened. The construction of the path follows giving preference to closing movements. The criterion shown in Figure 5 is used to accept these movements and the new solutions are used to update the set of no-dominated solutions *NDS*. Figure 7 shows the algorithm for the combination method.

[Figure 7 goes about here]

5. Computational Evaluation

The specific goals accomplished by the experiments are as follows. Firstly, to solve relative small size instances with the exact method to have a reference to compare with the metaheuristic algorithm. Also, a variation of the exact method was used to obtain approximate efficient sets for the comparison in larger instances. The variation consisted in running the optimization software for each point of the epsilon-constraint algorithm with a time limit of 3600 seconds. These approximate efficient sets were compared with those obtained with the metaheuristic algorithm to determine their quality, and the computational run times were compared to evaluate the efficiency of the metaheuristic algorithm.

To perform the computational study, instances of different sizes were randomly generated as described in Olivares-Benitez et al. (2012). The reader is invited to consult that work to have the details of the parameters generation. The sizes generated are shown in Table 1, where the group code indicates: [number of plants - number of potential distribution centers - number of customers - number of arcs between nodes].

[Table 1 goes about here]

5.1 True efficient sets

The "Backward epsilon-constraint method with estimated lower limit for f_2" (ReC) algorithm was used to solve the generated instances (Olivares-Benitez et al., 2012). The procedure was coded in C and compiled with Visual Studio 6.0. The CPLEX 11.1 callable library (ILOG SA, 2008) was used to solve optimally the sub-problems involved in the epsilon-constraint based algorithm. These routines were run in a 3.0 GHz, 1.0 Gb RAM, Intel Pentium 4 PC. The true efficient sets of the small instances of groups 5-5-5-2, 5-5-5-5, and 5-5-20-2 were obtained. The run times were recorded for comparison with the metaheuristic algorithm.

Figure 8 shows the efficient frontier for the instance number 2 of the group 5-5-5-5. The efficient frontier for the rest of the mentioned instances is similar. The points are not connected because of the discretization of time units. It is evident the tradeoff between cost (f_1) and time (f_2).

[Figure 8 goes about here]

5.2 Approximate efficient sets using the epsilon-constraint based algorithm

To have a comparison for large instances, the ReC algorithm was used with a time limit of 3600 seconds per each value of ε. The CPLEX 9.1 callable library (ILOG SA, 2005) was used to solve optimally the sub-problems involved in the epsilon-constraint based algorithm. These routines were run in a 3.0 GHz, 1.0 Gb RAM, Intel Pentium 4 PC. The approximate efficient sets and the run times were recorded for comparison with the metaheuristic algorithm.

5.3 Approximate efficient sets using the metaheuristic algorithm

The metaheuristic algorithm was coded in C. The CPLEX 9.1 callable library (ILOG SA, 2005) was used to solve the GAP and TP sub-problems generated within the algorithm. The algorithm was run in a 3.0 GHz, 1.0 Gb RAM, Intel Pentium 4 PC. The number of constructed solutions *NCS* in the metaheuristic algorithm was set to 100 solutions. The number of iterations before the execution of the combination method was set to 10.

5.4 Comparisons

To make comparisons of the efficient frontiers obtained with the algorithms several metrics were used. The computing time and the number of non-dominated points $|S_i|$ are reported. The ratio R_{POS} (S_i) (Altiparmak et al., 2006) is calculated also. This ratio is able to compare more than two efficient sets. To make the computations, a reference efficient set P must be constructed with the union of the efficient solutions of all the r sets, and the dominated solutions are eliminated. This metric indicates the ratio of points from the set S_i that belong to the reference efficient set P. A higher value of this metric is better, indicating the quality of the approximate efficient set obtained.

Additionally, based on the features of the problem treated in this work, a special metric was designed, although the principle may be adapted to other bi-objective combinatorial optimization problems. The discretization of objective f_2 and the number of objectives allows proceeding as follows for a pair of sets S_1 and S_2. A set of values T is constructed with each value of objective f_2 where values for objective f_1 exist in both sets:

$$T = \left\{ f_2(s) \vee f_2(s'), s \in S_1, s' \in S_2 \middle| \exists f_1(s) \wedge \exists f_1(s') \wedge f_2(s) = f_2(s') \right\}$$

Then an average deviation D_{ave} is calculated with the ratios of objective f_1 for each value of f_2 in the set T, as shown in equation (25).

$$D_{ave} = \frac{\sum_{t \in T} \frac{f_1(s) : f_2(s) = t}{f_1(s') : f_2(s') = t}}{|T|} \qquad \forall\, s \in S_1,\, s' \in S_2 \qquad (25)$$

The idea is very simple. For a fixed value of objective f_2 the ratio $f_1\,(s)\,/\,f_1\,(s')$ is calculated only if the values of objective f_1 are available in both sets. Then the average of these ratios is calculated. The minimum D_{min} of these ratios is calculated with equation (26).

$$D_{\min} = \min_{t \in T}\left(\frac{f_1(s) : f_2(s) = t}{f_1(s') : f_2(s') = t}\right) \qquad \forall\, s \in S_1,\, s' \in S_2 \qquad (26)$$

For these metrics D_{ave} and D_{min}, the true efficient sets and the approximate efficient sets obtained with the ReC algorithm take place in the computations as S_2, and the approximate efficient sets obtained with the metaheuristic algorithm are considered as S_1.

Table 2 and Table 3 show the results for five instances of each size. The results of the epsilon-constraint based algorithm are identified with the code [*ReC*] and the results of the metaheuristic algorithm are identified with the code [*MH*]. Table 2 presents the comparison between the exact method and the metaheuristic method, i.e. the true efficient sets and the approximate efficient sets respectively. The results in Table 3 compare the performance of the exact method with time limit and the metaheuristic method, i.e. approximate efficient sets in both cases.

The comparison of results for each metric must be made as follows. A greater value for $|S_i|$ and $R_{POS}\,(S_i)$ is better. These values indicate the size and quality of the efficient frontier. A lower value, less than or equal to 1.0, for metrics D_{min} and D_{ave} indicates that the metaheuristic algorithm achieves lower cost (f_1) compared to the epsilon-constraint based algorithm, for the same transportation time (f_2).

[Table 2 goes about here]

[Table 3 goes about here]

A visual comparison of the efficient frontiers is shown in Figures 9 and 10 for a small instance and a very large instance respectively. It is evident the trade-off between the cost and time objectives, with best times for worse costs. It can be observed in Figure 9 that for small instances, the metaheuristic algorithm got very close to the efficient frontier and in some points achieved the true efficient solution. For large instances, like in the case of Figure 10, the metaheuristic algorithm obtained a better efficient frontier than the epsilon-constraint based algorithm. It was observed in several experiments that obtaining a solution for middle values of f_2 (time objective) was more difficult, which explains the form of the frontier obtained with the epsilon-constraint based algorithm. Both algorithms achieved good solutions when the problem approaches to the single-objective case in the extremes of the efficient frontier.

[Figure 9 goes about here]

[Figure 10 goes about here]

In particular, for the instances of group 5-5-5-2 the total run time for the metaheuristic algorithm was greater than the time for the epsilon-constraint based algorithm. In this case the instances are very small and the exact method can obtain true efficient sets easily. However, the metaheuristic algorithm spent more time in useless exploration of the solutions space without achieving good approximate frontiers.

For the instances of groups 50-50-50-2 and 50-50-100-2 the time of the metaheuristic algorithm was greater than the time for the epsilon-constraint based algorithm. The explanation here is different. In this case the epsilon-constraint based algorithm ran with a time limit per each value of ε. In all the instances of Table 3 this time limit was reached in almost all the values of ε. The total time is very similar for these groups of instances because the number of points for ε values to be explored was also similar. This characteristic of the parameters come from the instance generation procedure described in Olivares-Benitez et al. (2012). In the case of the metaheuristic algorithm, the run time depends on the size of the instance and then the time is expected to grow for larger instances. The total run time in both algorithms was not controlled.

6 Conclusions

The process of supply chain design involves decisions over several aspects. The most treated decisions in the literature are facility location, transportation flows, production levels, supplier selection, and inventory levels. Nevertheless only the most recent works include transportation channel selection. The supply chain design problem addressed here incorporates the selection of the transportation channel that produces a cost-time tradeoff. Hence as a bi-objective problem, the solution is not unique and a set of efficient solutions must be obtained. The construction of a set of efficient solutions follows an *a posteriori* approach where the decision maker will take the final decision considering other criteria to select one among the different solutions obtained. As noted in the literature review section, this approach has not been addressed before in supply chain design although some works have considered it partially.

In this work we designed a metaheuristic algorithm *ad-hoc* to solve the problem treated. This metaheuristic incorporates elements from greedy functions, Scatter Search, Path Relinking and Mathematical Programming. It decomposes the construction of a solution in a hierarchy of decisions. Some of the steps require the use of mathematical programming software to solve a generalized assignment problem and a transportation problem. This approach has been formalized recently as “Matheuristics”, which combine metaheuristics and mathematical programming techniques (Maniezzo et al., 2010). The literature review showed that traditional metaheuristics like Genetic Algorithms and Tabu Search have been used frequently for single objective versions of supply chain design problems. Recently multiobjective versions of these metaheuristics and more sophisticated methods have been used for multiobjective supply chain design problems. However the matheuristic hybrid presented here is a novelty in the field of supply chain design. The use of this type of metaheuristic algorithms and the transportation channel selection in the context of multiobjective supply chain design represent major contributions of this paper.

The comparison in Table 2 showed that the metaheuristic algorithm becomes competitive in terms of computing time with the exact method for small instances, although the efficient frontiers have lower cardinality and quality. However the approximate efficient sets obtained with the metaheuristic algorithm are not too far from those obtained with the exact method as was observed in the example of Figure 9.

For large instances, the results in Table 3 proved that the metaheuristic algorithm becomes competitive in the three metrics of comparison: computing time, cardinality and quality of the efficient sets obtained. We believe that a great benefit comes from integrating mathematical programming commercial software into the algorithm to solve the transportation and generalized assignment sub-problems. At the same time, the population-based approach of the metaheuristic makes a good exploration of the solutions space achieving well distributed efficient frontiers.

The model presented here may be used for the design of supply chains of products with low complexity or products with few components. The application may be in short supply chains where lead time may be a competitive advantage or a requisite. These characteristics apply well to perishable or seasonal products. Examples of these products can be found in the food industry, pharmaceutical industry, chemical industry, and apparel industry. The metaheuristic algorithm developed in this work delivers several alternatives in the efficient frontier for the decision maker. It also obtains solutions relatively fast, considering that the problem involves strategic decisions with impact in the mid-to long term. Therefore several scenarios can be analyzed easily with changing parameters of demand or costs considering uncertainty, before taking a final decision.

The following are limitations and consequently possible extensions to the model. The model may be extended to include multiple commodities, direct flows from plants to customers, and intra-echelon flows. Also routing decisions, technology selection, capacity levels and international supply chain aspects may be considered. The transportation time has an effect in the size of pipeline inventory and safety stock inventory that may be considered into the cost objective function. Recent works in multiobjective supply chain design include environmental and risk criteria in parallel to economic objectives. The explicit modeling of uncertainty of the demand and other parameters or a multi-period approach may also be addressed. Nevertheless, these elements change the structure of the problem and a major modification of the metaheuristic algorithm should be done.

The results of the metaheuristic algorithm were compared favorably to the results of an epsilon-constraint based algorithm. However it may be interesting the comparison of the metaheuristic algorithm with other methods. The natural candidates for this additional comparison are evolutionary algorithms like SPEA 2 (Zitzler et al., 2001) and NSGA-II (Deb et al., 2002). Moreover, considering the structure of the epsilon-constrained model, methods like decomposition schemes, Lagrangian relaxation, and single-objective metaheuristics may be implemented in a sequential algorithm to obtain the approximate efficient frontier.

Acknowledgements: This research has been supported by ITESM Research Fund CAT128, UPAEP Research Funds, and the Mexican National Council for Science and Technology (grant SEP-CONACYT 61903). The authors thank the anonymous referees for their helpful comments and suggestions.

References

Aikens, C.H., 1985. Facility location models for distribution planning. European Journal of Operational Research, 22(3), 263-279.

Altiparmak, F., Gen, M., Lin, L., Paksoy, T., 2006. A genetic algorithm approach for multi-objective optimization of supply chain networks. Computers and Industrial Engineering, 51(1), 197-216.

Arntzen, B.C., Brown, G.C., Harrison, T.P., Trafton, L.L., 1995. Global supply chain management at Digital Equipment Corporation. Interfaces, 25(1), 69-93.

Beamon, B., 1998, Supply chain design and analysis: Models and methods. International Journal of Production Economics, 55(3), 281-294.

Chaabane, A., Ramudhin, A., Paquet, M., 2012. Design of sustainable supply chains under the emission trading scheme. International Journal of Production Economics, 135(1), 37-49.

Chan, F.T.S., Chung, S.H., Choy, K.L., 2006. Optimization of order fulfillment in distribution network problems. Journal of Intelligent Manufacturing, 17(3), 307-319.

Cornuejols, G., Nemhauser, G.L., Wolsey, L.A., 1990. The uncapacitated facility location problem. In: Mirchandani, P.B., Francis, R.L. (Eds.), Discrete Location Theory, Chapter 3. Wiley, New York, USA, 119-171.

Current, J., Min, H., Schilling, D., 1990. Multiobjective analysis of facility location decisions. European Journal of Operational Research, 49(3), 295-307.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2), 182-197.

Ehrgott, M., 2005. Multicriteria Optimization. Springer, Berlin, Germany.

ElMaraghy, H.A., Majety, R., 2008. Integrated supply chain design using multi-criteria optimization. International Journal of Advanced Manufacturing Technology, 37(3), 371-399.

Farahani, R.Z., SteadieSeifi, M., Asgari N., 2010. Multiple criteria facility location problems: A survey. Applied Mathematical Modelling, 34(7), 1689-1709.

Funaki, K., 2012. Strategic safety stock placement in supply chain design with due-date based demand. International Journal of Production Economics, 135(1), 4-13.

Graves, S.C., Willems, S.P., 2005. Optimizing the supply chain configuration for new products. Management Science, 51(8), 1165-1180.

Griffis, S.E., Bell, J.E., Closs, D.J., 2012. Metaheuristics in Logistics and Supply Chain Management. Journal of Business Logistics, 33(2), 90-106.

Gumus, A.T., Guneri A.F., Keles, S., 2009. Supply chain network design using an integrated neuro-fuzzy and MILP approach: A comparative design study. Expert Systems with Applications, 36(10), 12570-12577.

ILOG SA, 2005. ILOG CPLEX Callable Library C API 9.1 Reference Manual. ILOG, France.

ILOG SA, 2008. ILOG CPLEX Callable Library C API 11.1 Reference Manual. ILOG, France.

Klose, A., Drexl, A., 2005. Facility location models for distribution system design. European Journal of Operational Research, 162(1), 4-29.

Laguna, M., Marti, R., 2003. Scatter Search: Methodology and Implementations in C. Kluwer Academic Publishers, Norwell, USA.

Liao, S.-H., Hsieh, C.-L., Lai, P.-J., 2011. An evolutionary approach for multi-objective optimization of the integrated location–inventory distribution network problem in vendor-managed inventory. Expert Systems with Applications, 38(6), 6768-6776.

Maniezzo, V., Stutzle, T., Voß, S., 2010. Matheuristics: Hybridizing metaheuristics and mathematical programming. Annals of Information Systems, Vol. 10. Springer, New York, USA.

Mansouri, S.A., Gallear, D., Askariazad, M.H., 2012. Decision support for build-to-order supply chain management through multi-objective optimization. International Journal of Production Economics, 135(1), 24-36.

Marvin, W.A., Schmidt, L.D., Benjaafar, S., Tiffany, D.G., Daoutidis, P., 2012. Economic Optimization of a Lignocellulosic Biomass-to-Ethanol Supply Chain. Chemical Engineering Science, 67(1), 68-79.

Marti, R., Campos, V., Resende, M.G.C., Duarte, A., 2011. Multiobjective GRASP with Path Relinking. Technical Report, AT&T Labs Research, Florham Park, New Jersey, USA.

Melo, M.T., Nickel, S., Saldanha-da-Gama, F., 2009. Facility location and supply chain management – A review. European Journal of Operational Research, 196(2), 401-412.

Moncayo-Martinez, L.A., Zhang, D.Z., 2011. Multi-objective ant colony optimisation: A meta-heuristic approach to supply chain design. International Journal of Production Economics, 131(1), 407-420.

Olivares-Benitez, E., González-Velarde, J.L., Ríos-Mercado, R.Z., 2012. A supply chain design problem with facility location and bi-objective transportation choices. TOP, 20(3), 729-753.

Paksoy, T., Pehlivan, N.Y., Özceylan, E., 2012. Application of fuzzy optimization to a supply chain network design: A case study of an edible vegetable oils manufacturer. Applied Mathematical Modelling, 36(6), 2762-2776.

Pati, R.K., Vrat, P., Kumar, P., 2008. A goal programming model for paper recycling system. Omega, 36(3), 405-417.

Pinto-Varela, T., Barbosa-Póvoa, A.P.F.D., Novais, A.Q., 2011. Bi-objective optimization approach to the design and planning of supply chains: Economic versus environmental performances. Computers and Chemical Engineering, 35(8), 1454-1468.

Pishvaee, M.S., Farahani, R.Z., Dullaert, W., 2010. A memetic algorithm for bi-objective integrated forward/reverse logistics network design. Computers and Operations Research, 37(6), 1100-1112.

Resende, M.G.C., Ribeiro C.C., 2003. Greedy Randomized Adaptive Search Procedures. In: Glover, F., Kochenberger, G.A. (Eds.), handbook of Metaheuristics, Chapter 8. Kluwer Academic Publishers, New York, USA, 219-250.

Sadjady, H., Davoudpour, H., 2012. Two-echelon, multi-commodity supply chain network design with mode selection, lead-times and inventory costs. Computers and Operations Research, 39(7),1345-1354.

Sahin, G., Sural, H., 2007. A review of hierarchical facility location models. Computers and Operations Research, 34(8), 2310-2331.

Sousa, R., Shah, N., Papageorgiou, L.G., 2008. Supply chain design and multilevel planning - An industrial case. Computers and Chemical Engineering, 32(11), 2643–2663.

Steuer, R.E., 1989. Multiple Criteria Optimization: Theory, Computation and Application. Krieger Publishing Company, Malabar, USA.

Talbi, E., Metaheuristics: From Design to Implementation. Wiley, Hoboken, USA.

Thomas, D.J., Griffin, P.M., 1996. Coordinated supply chain management. European Journal of Operational Research, 94(1), 1-15.

Vidal, C.J., Goetschalckx, M., 1997. Strategic production-distribution models: A critical review with emphasis on global supply chain models. European Journal of Operational Research, 98(1), 1-18.

Zeng, D.D., 1998. Multi-issue Decision Making in Supply Chain Management and Electronic Commerce. PhD Dissertation, Graduate School of Industrial Administration and Robotics Institute, Carnegie Mellon University, Pittsburgh, USA, December.

Zitzler, E., Laumanns, M., Thiele, L., 2001. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

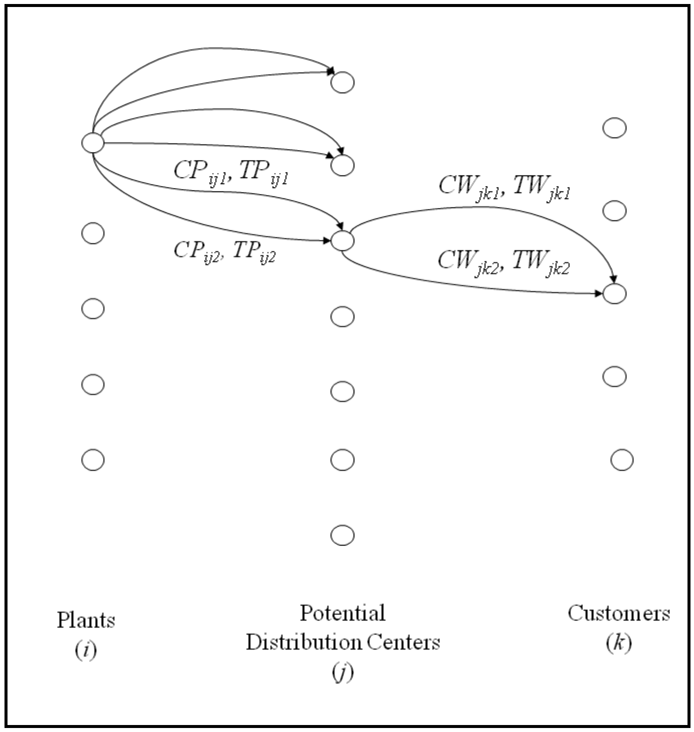

Figure 1 Single product, single period, and two-echelon distribution system. Each transportation channel has a time (TP_{ijl}, TW_{jkl}) and a unitary cost (CP_{ijl}, CW_{jkl}) associated. Source: Olivares-Benitez et al. (2012).

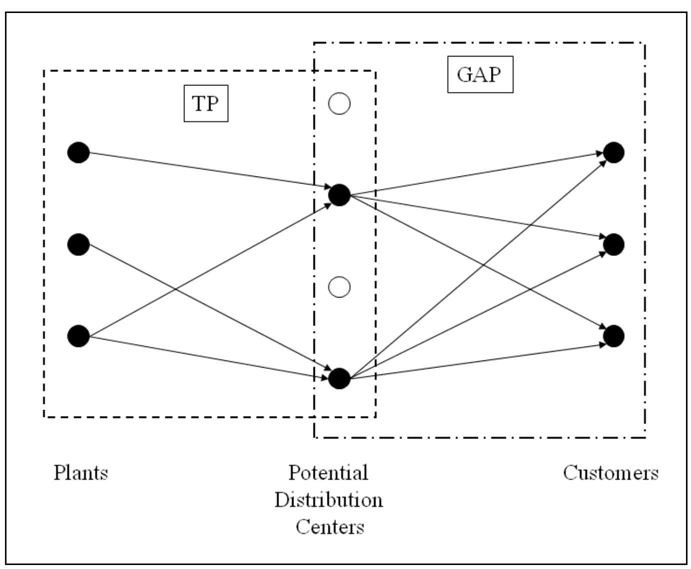

Figure 2 Generalized Assignment Problem (GAP) and Transportation Problem (TP) in the hierarchical construction procedure.

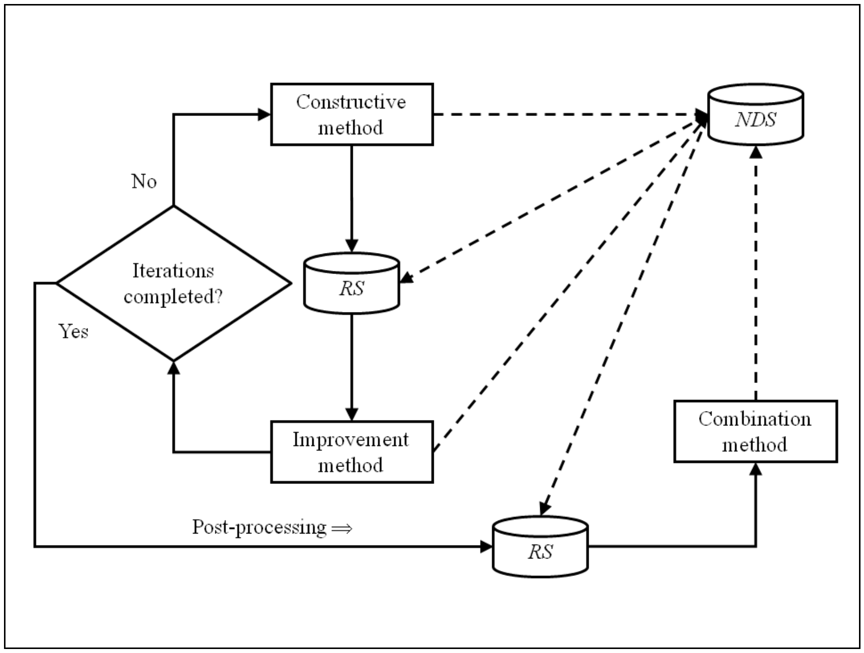

Figure 3 Scheme of the metaheuristic algorithm.

Method **Constructive**
Input: Instance data; Number of Constructed Solutions (*NCS*).
Output: Set of Constructed Solutions, $CS = \{s^r \mid r = 1,\ldots, NCS\}$ or general infeasibility message.
<u>BEGIN</u>
01. Check general feasibility of the instance under the following conditions:

$$\sum_{i\in I} MP_i \geq \sum_{k\in K} D_k\ ,\ \sum_{j\in J} MW_j \geq \sum_{k\in K} D_k$$

02. If the instance is infeasible:
03. Return message of infeasibility.
04. Else:
05. $CS = \varnothing$.
06. For $r = 1, \ldots, NCS$:
07. Initialize $Z_j = 0$, $A_{ijl} = 0$, $B_{jkl} = 0$, $i \in I, j \in J, k \in K, l \in LP_{ij}, l \in LW_{jk}$.
08. Initialize s^r is incomplete.
09. Calculate the vector $[\lambda_c^r, \lambda_t^r]$ for solution s^r.
10. Calculate the aggregated greedy function for each element $\phi(arc_{ijl}), \phi(arc_{jkl})$ using equations (20) – (21).
11. While solution s^r is incomplete and the instance is feasible:
12. While $\sum_{j\in J} Z_j MW_j < \sum_{k\in K} D_k$:
13. Select randomly a distribution center $j' \in J$, $Z_{j'} = 1$.
14. End While.
15. Set of open distribution centers $J' = \{j \in J \mid Z_j = 1\}$.
16. s^r = **Hierarchical construction procedure** (*J'*).
17. If s^r is infeasible:
18. If $|J'| < |J|$:
19. Go To Step 13 to open another distribution center.
20. Else:
21. Return a message of infeasibility for the instance.
22. End If.
23. Else:
24. $CS = CS \cup \{s^r\}$ and the associated vector $[\lambda_c^r, \lambda_t^r]$ is stored in the structure of the solution s^r.
25. End If.
26. End While.
27. End For.
28. Return the set *CS* in the output file.
29. End If
<u>END</u>

Figure 4 Algorithm for the constructive method.

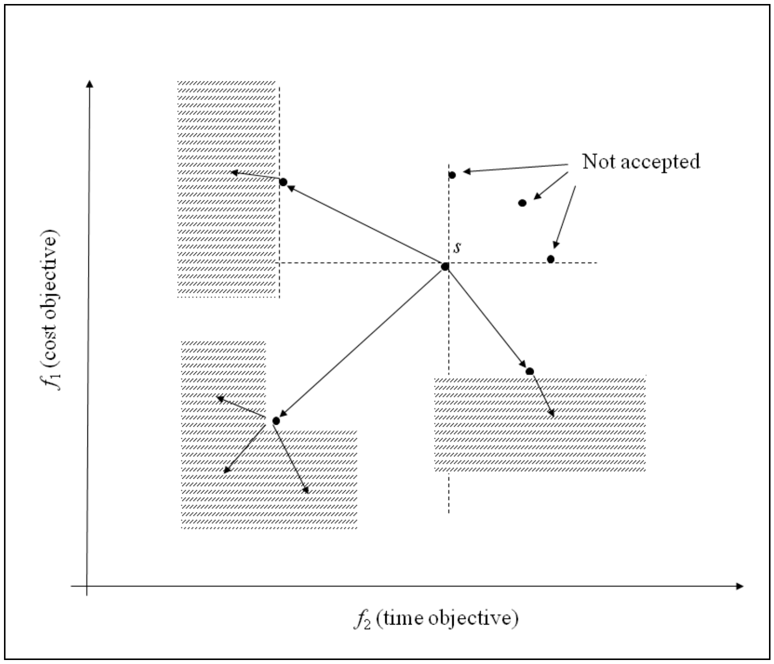

Figure 5 Scheme of the acceptance criterion and direction of improvement.

Method **Improvement**
Input: Instance data; Reference set of solutions *RS*; Approximate efficient set *NDS*.
Output: Approximate efficient set *NDS* updated.
BEGIN
For each $s \in RS$:
    Current solution $s' = s$.
    Exit_local_search = 0.
    While *Exit_local_search* = 0:
        Initialize the set of improved solutions $IS = \varnothing$.
        Obtain solution s^c or infeasibility message exploring the **closing neighborhood** $CN(s')$.
        NDS = **Update NDS set** (s^c, NDS).
        If s^c meets the **acceptance criterion** and **direction of improvement**, $IS = IS \cup \{s^c\}$.
        Obtain solution s^o or infeasibility message exploring the **opening neighborhood** $ON(s')$.
        NDS = **Update NDS set** (s^o, NDS).
        If s^o meets the **acceptance criterion** and **direction of improvement**, $IS = IS \cup \{s^o\}$.
        Obtain solution s^e or infeasibility message exploring the **exchange neighborhood** $EN(s')$.
        NDS = **Update NDS set** (s^e, NDS).
        If s^e meets the **acceptance criterion** and **direction of improvement**, $IS = IS \cup \{s^e\}$.
        If $IS \neq \varnothing$:
            Select randomly a solution $\hat{s} \in IS$
            New current solution $s' = \hat{s}$.
        Else:
            Exit_local_search = 1.
        End If.
    End While.
End For.
END

Figure 6 Algorithm for the improvement method.

Method **Combination**

Input: Instance data; Reference set of solutions *RS*; Approximate efficient set *NDS*.

Output: Approximate efficient set *NDS* updated.

BEGIN

01. For ∀ (initiating solution) $s \in RS$:

02. For ∀ (guiding solution) $r \in RS$:

03. If $s \neq r$ and $Z_j^s \neq Z_j^r \; \forall j \in J$:

04. Create sets *FC* and *FO*; Sort sets $FC = \{j_1, j_2, \ldots \mid \phi(dc_{j_i}) \geq \phi(dc_{j_{i+1}})\}$ and $FO = \{j_1, j_2, \ldots \mid \phi(dc_{j_i}) \leq \phi(dc_{j_{i+1}})\}$.

05. Create intermediate solution q, $Z_j^q = Z_j^s \; \forall j \in J$, $\lambda_c^q = \lambda_c^r$, $\lambda_t^q = \lambda_t^r$.

06. If partial solution q is feasible (based on accumulated capacity):

07. Use the **Hierarchical construction procedure** to complete solution q.

08. If complete solution q is feasible:

09. *NDS* = **Update NDS set** (q, *NDS*).

10. End If.

11. End If.

12. $n = 1$, $p = 1$.

13. While $n \leq |FC|$:

14. Modify intermediate solution q making $Z_{j_n}^q = 0$, $j_n \in FC$.

15. If partial solution q is feasible (based on accumulated capacity):

16. Use the **Hierarchical construction procedure** to complete solution q.

17. If complete solution q is feasible:

18. *NDS* = **Update NDS set** (q, *NDS*); $n = n + 1$; Go To Step 13.

19. Else:

20. $n = n + 1$; Go To Step 26.

21. End If.

22. Else:

23. $n = n + 1$; Go To Step 26.

24. End If.

25. End While.

26. While $p \leq |FO|$:

27. Modify intermediate solution q making $Z_{j_p}^q = 0$, $j_p \in FO$.

28. If partial solution q is feasible (based on accumulated capacity):

29. Use the **Hierarchical construction procedure** to complete solution q.

30. If complete solution q is feasible:

31. *NDS* = **Update NDS set** (q, *NDS*); $p = p + 1$; Go To Step 13.

32. Else:

33. $p = p + 1$; Go To Step 26.

34. End If

35. Else:

36. $p = p + 1$; Go To Step 26.

37. End If.

38. End While.

39. End If.

40. End For.

41. End For.

END

Figure 7 Algorithm for the combination method.

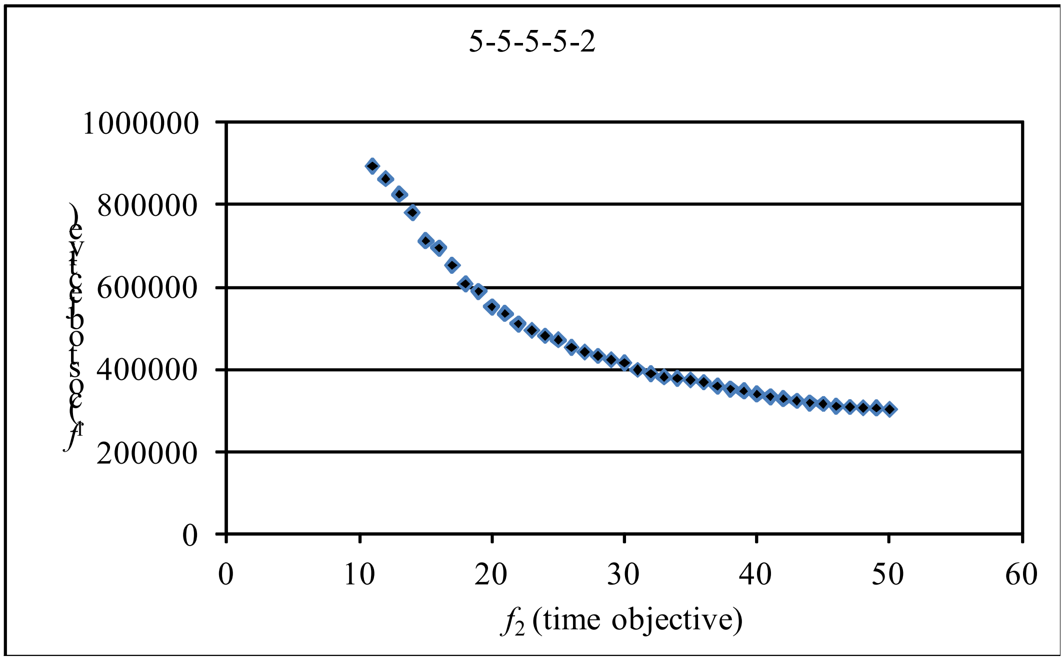

Figure 8 Set of non-dominated points for instance number 2 of group 5-5-5-5. Source: Olivares-Benitez et al. (2012).

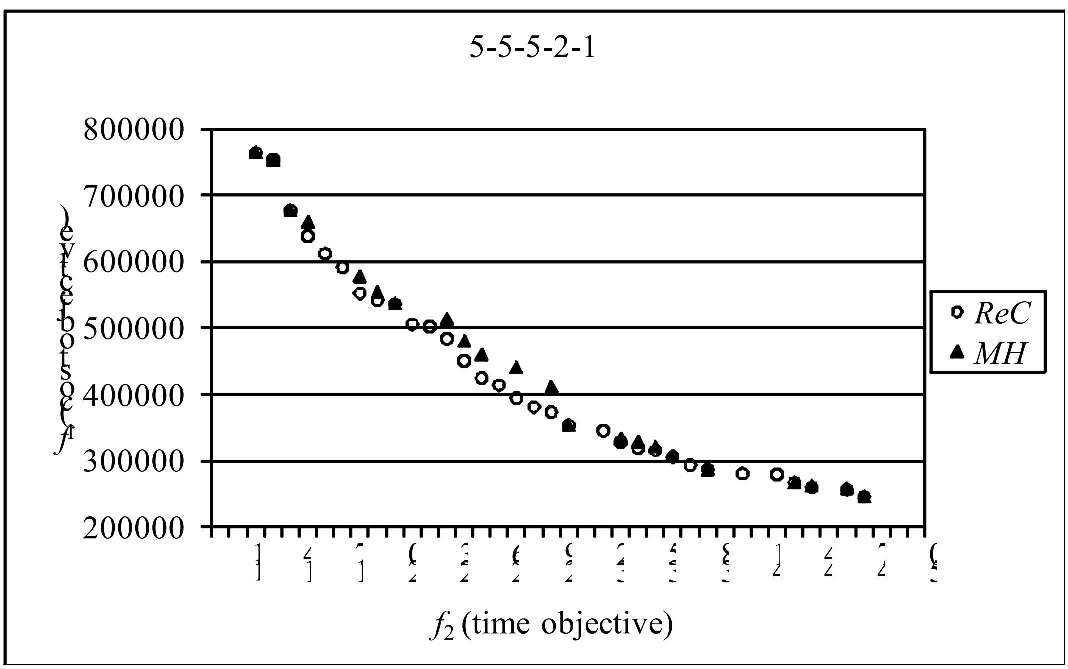

Figure 9 Comparison of the efficient frontiers for instance number 1 of group 5-5-5-2.

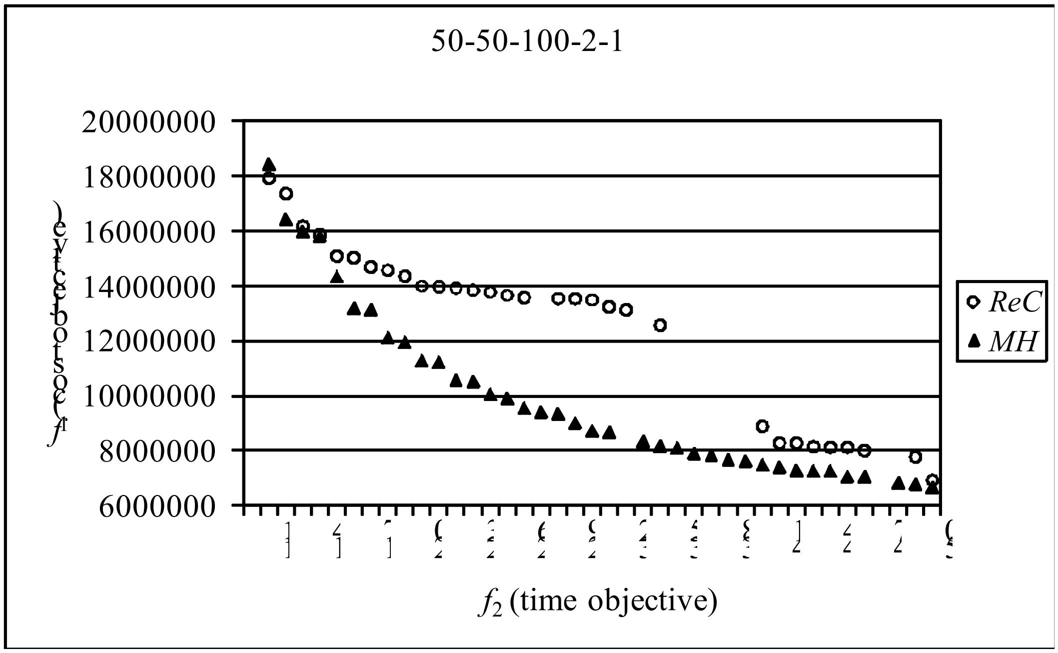

Figure 10 Comparison of the approximate efficient frontiers for instance number 1 of group 50-50-100-2.

Table 1 Generated instances.

Group code	Number of instances	Number of binary variables	Number of constraints
5-5-5-2	5	105	385
5-5-5-5	5	255	835
5-5-20-2	5	255	940
5-20-20-2	5	1020	3625
20-20-20-2	5	1620	5740
20-20-20-5	5	4020	12940
20-20-50-5	5	7020	22600
50-50-50-2	5	10050	35350
50-50-100-2	5	15050	52950

Group code indicates: [number of plants - number of potential distribution centers - number of customers - number of arcs between nodes]

Table 2 Comparison of results from the metaheuristic algorithm [*MH*] and the epsilon-constraint based algorithm [*ReC*] for small instances.

Group code	Instance	Total time (sec) [*MH*]	Total time (sec) [*ReC*]	$\lvert S_{ReC}\rvert$	$R_{POS}(ReC)$	$\lvert S_{MH}\rvert$	$R_{POS}(MH)$	D_{ave}	D_{min}
5-5-20-2	1	29	236	31	1.000	20	0.050	1.042	1.000
5-5-20-2	2	33	269	33	1.000	20	0.050	1.031	1.000
5-5-20-2	3	71	452	33	1.000	22	0.045	1.045	1.000
5-5-20-2	4	54	324	32	1.000	20	0.050	1.052	1.000
5-5-20-2	5	74	491	33	1.000	27	0.037	1.028	1.000
5-5-5-5	1	92	134	38	1.000	32	0.125	1.020	1.000
5-5-5-5	2	64	159	40	1.000	25	0.160	1.027	1.000
5-5-5-5	3	63	219	39	1.000	27	0.259	1.014	1.000
5-5-5-5	4	147	180	39	1.000	31	0.194	1.022	1.000
5-5-5-5	5	64	111	39	1.000	28	0.179	1.018	1.000
5-5-5-2	1	75	7	32	1.000	22	0.364	1.028	1.000
5-5-5-2	2	36	11	29	1.000	21	0.095	1.024	1.000
5-5-5-2	3	43	12	28	1.000	22	0.364	1.019	1.000
5-5-5-2	4	37	24	31	1.000	17	0.412	1.021	1.000
5-5-5-2	5	41	10	25	1.000	18	0.389	1.017	1.000

Group code indicates: [number of plants - number of potential distribution centers - number of customers - number of arcs between nodes]

Table 3 Comparison of results from the metaheuristic algorithm [*MH*] and the epsilon-constraint based algorithm with time limit [*ReC*] for large instances.

Group code	Instance	Total time (sec) [*MH*]	Total time (sec) [*ReC*]	$\lvert S_{ReC}\rvert$	$R_{POS}(ReC)$	$\lvert S_{MH}\rvert$	$R_{POS}(MH)$	D_{ave}	D_{min}
50-50-100-2	1	53715	24022	31	0.032	38	0.974	0.831	0.646
50-50-100-2	2	59076	24022	30	0.000	37	1.000	0.798	0.645
50-50-100-2	3	55026	24026	37	0.081	37	1.000	0.816	0.602
50-50-100-2	4	57049	24604	33	0.091	37	0.946	0.859	0.681
50-50-100-2	5	45386	24020	37	0.027	38	0.974	0.810	0.643
50-50-50-2	1	32901	24604	39	0.051	37	0.973	0.903	0.813
50-50-50-2	2	34144	24604	39	0.077	40	0.950	0.888	0.795
50-50-50-2	3	41621	24010	37	0.027	36	0.972	0.850	0.698
50-50-50-2	4	27755	24010	39	0.026	39	0.974	0.874	0.780
50-50-50-2	5	30655	24008	36	0.028	40	0.975	0.909	0.843
20-20-50-5	1	17756	24603	37	0.054	39	0.949	0.912	0.800
20-20-50-5	2	20145	24603	41	0.024	41	0.976	0.899	0.793
20-20-50-5	3	21887	24007	39	0.026	37	0.973	0.898	0.799
20-20-50-5	4	18764	24603	40	0.025	38	0.974	0.908	0.816
20-20-50-5	5	18001	24010	40	0.100	37	0.973	0.908	0.835
20-20-20-5	1	5029	24270	41	0.049	41	0.951	0.927	0.842
20-20-20-5	2	5426	24487	40	0.050	40	0.975	0.929	0.860
20-20-20-5	3	3597	24009	39	0.077	39	0.949	0.930	0.844
20-20-20-5	4	2764	24007	41	0.049	40	0.975	0.924	0.867
20-20-20-5	5	5209	24605	38	0.053	41	0.951	0.936	0.859
20-20-20-2	1	4680	22937	40	0.125	38	0.921	0.967	0.900
20-20-20-2	2	4100	23405	39	0.128	39	0.872	0.965	0.906
20-20-20-2	3	2847	23022	40	0.200	38	0.842	0.962	0.888
20-20-20-2	4	4238	23407	40	0.150	39	0.872	0.973	0.878
20-20-20-2	5	4612	23446	39	0.205	39	0.821	0.979	0.915
5-20-20-2	1	3615	22257	38	0.289	37	0.703	0.973	0.900
5-20-20-2	2	3097	22257	38	0.289	39	0.718	0.977	0.905
5-20-20-2	3	2346	22231	38	0.368	37	0.649	0.983	0.914
5-20-20-2	4	2403	21709	39	0.282	39	0.718	0.981	0.899
5-20-20-2	5	4425	21669	39	0.282	39	0.718	0.977	0.920

Group code indicates: [number of plants - number of potential distribution centers - number of customers - number of arcs between nodes]