A Constructive Procedure for Finding Good Starting Solutions to the Network Design Problem with Uncertain Parameters

FERNANDO PÉREZ, ADA M. ALVAREZ AND KARIM DE ALBA
Facultad de Ingeniería Mecánica y Eléctrica,
Universidad Autónoma de Nuevo León,
MONTERREY, NL, MÉXICO

Abstract:- Given a network with a set of nodes and a set of potential edges, we must decide what edges will be included in the design to satisfy demands between pairs of nodes. Each edge has associated a finite capacity, a fixed cost and routing costs. Uncertainty will be considered in important input parameters such as demands and routing costs and will be modeled through scenarios. The objective is to find a network design that is "good" across all scenarios potentially realizable. At the present time, there exists no exact algorithm that can solve large instances, common in several applications, in a reasonable period of time. A GRASP is proposed that gives solutions of acceptable quality.

Key-Words: Robust optimization, Multicommodity network, Capacitated network design, Uncertain data, Heuristics.

1 Introduction

The network design problem consists of selecting, from an available set of potential edges, a subset of them which should be capable of transporting the demands of various goods. The selection process involves the minimization of the incurred costs (selection and routing costs).

One common assumption in this kind of problem concerns the capacity of the edges, which can be considered finite or infinite. If capacity is considered finite, the problem becomes more difficult to solve. Other important assumption relates to the certainty or uncertainty of key problem parameters.

Interesting results have been derived for the deterministic version of the problem, [1, 2, 4, 7], that is, when all parameters are considered known, but less effort has been directed towards the problem that manage uncertainty in some parameters, which is more realistic but harder and poses considerable algorithmic challenges.

A common approach is to capture uncertainty using scenarios and formulating a deterministic equivalent problem, but it is well known that the solutions to such "worst-case" or "mean-value" problems are inadequate.

It is also possible to know (or to estimate) the probability of occurrence of the random parameters and then use a stochastic optimization model to solve it [9]. The stochastic optimization approach recognizes the presence of multiple data instances that might be potentially realized in the future and typically the model will attempt to generate a decision that optimizes an expected performance measure.

Nevertheless, the decision makers know that for any potentially realizable scenario they have to live with the consequences on system performance of the decision made. Risk averse decision makers are reasonable more interested in hedging against the risk of poor system performance for some realizations of data scenarios than in optimizing expected system performance over all potential scenarios or just performance of the most likely scenario. This is particularly important for decisions of unique nature and thus encountered only once. Thus the performance of a decision across all potentially realizable scenarios is important. A decision will be "robust" with respect to optimality if it remains "close" to optimal for any realization of the input data scenarios [8,12]. It has been used in diverse applications as the capacitated international sourcing problem [10], capacity expansion in telecommunications networks [11], etc.

We tackle here a multicommodity capacitated network design problem on non-directed networks. Each edge has a finite capacity shared among every commodity using that edge regardless of the direction of flow. Uncertainty will be considered on demands of each

commodity and on the transportation costs, and will be modeled via scenarios.
Gutiérrez, Kouvelis and Kurawarwala [6] address the uncapacitated version of the multicommodity network design problem. Taking into account that there exists no exact algorithm that can solve even medium-size instances, we propose a constructive procedure based on GRASP to find a set of "good" initial solutions to the problem addressed here.

2 Problem Formulation

Let $G=(N, A)$ be the undirected graph underlying the network to be designed, so we have:
- N : set of nodes.
- A : set of available edges.
- A' : set of arcs in the network. (For each $\{i, j\} \in A$ there are two arcs in A' : (i , j) and (j , i))
- K: set of commodities. For each $k \in K$, $O(k)$ and $D(k)$ are their origin and destination points respectively.
- f_{ij} :fixed cost of including edge {i,j} in the design.
- u_{ij} :capacity of edge{i,j},which must be shared by all the commodities moving through it in any direction.

The uncertainty is modeled in the demand of each commodity and in the transportation costs via a set of scenarios S
- d^s_k : demand of commodity k under scenario $s \in S$.
- c^s_{ijk} : unit costs of sending commodity k through arc (i, j), under scenario $s \in S$.
- p_s the probability of occurrence of scenario s.

The model has two types of decision variables. The first type is a binary variable that models the design selection, and is defined as y_{ij} = 1, if edge $\{i, j\}$ is included in the network design, y_{ij} = 0, otherwise. The second type, which will be denoted by x^s_{ijk} is a continuous variable representing the amount of flow of commodity k going through arc (i, j) under scenario s. Note that, although the network is undirected, the flow is directed.
Suppose that y^* is a set of values for the y variables and that z_s is the optimal objective value associated with the following multicommodity network flow problem defined for each scenario s :

$$Min\ z_s = \sum_{k \in K} \sum_{(i,j) \in A} c^s_{ijk}\ x^s_{ijk} \qquad \textbf{(1)}$$

$$\sum_{\{j:(i,j)\in A\}} x^s_{ijk} - \sum_{\{j:(j,i)\in A\}} x^s_{ijk} = \begin{cases} d^s_k \text{ if } i = O(k) \\ -d^s_k \text{ if } i = D(k) \\ 0 \text{ other} \end{cases} \qquad \textbf{(2)}$$

$$\sum_{k \in K} (x^s_{ijk} + x^s_{jik}) \le u_{ij} y^*_{ij} \qquad \forall \{i, j\} \in A \qquad \textbf{(3)}$$

$$x^s_{ijk} \ge 0 \qquad \forall k \in K, \forall (i, j) \in A' \qquad \textbf{(4)}$$

Constraints (2) are the standard flow conservation constraints, which are imposed for each commodity k and each node i. Inequalities (3) ensure that the flow of all commodities through any direction of edge {i, j} does not exceed the capacity of this edge and prohibit flow through inactive edges (that is, edges not included in the design). The usual non-negativity constraints are set in (4).

Our objective is to find values for the y variables that minimizes the following function

$$\sum_{s \in S} p_s \left(\sum_{\{i,j\} \in A} f_{ij}\ y_{ij} + z_s \right) + \omega \sqrt{\frac{\sum_{s \in S^+} p_s \left(z_s - E(z_s)\right)^2}{\sum_{s \in S^+} p_s}} \qquad \textbf{(5)}$$

where:
$S^+ = \{s : z_s - E(z_s) \ge 0\}$

$E(z_s) = \sum_{s \in S} p_s\ z_s$

This means that our objective function penalizes only the positive deviations from the expected value, i.e. , we penalize those situations in which the objective value in a given scenario exceeds the expected cost. Note that ω is a factor that the decision maker can adjust to give more or less importance to the risk component of the objective function. The resulting model becomes a mixed non-linear program for which we have developed a heuristic procedure based on GRASP constructions.

3 Solution Method

Our solution method is based on GRASP constructions. GRASP, or Greedy Randomized Adaptive Search Procedure, is a metaheuristic that usually applies a randomized greedy constructive heuristic. Most GRASP implementations also include a

local search that is used to improve upon the solutions generated with the randomized greedy function. Details of the methodology and a survey of applications can be found in [13].

The design and implementation of heuristics requires an efficient computational representation of solutions. This requirement is accomplished here by splitting a solution into independent elements, one element by commodity, called *blocks of paths* and denoted by B^k. Then, for each commodity k whose demand will be transported on the network, a block B^k, formed by one or more paths selected from a list of shortest paths, is generated. Even though commodities may share the capacity of the edges, the blocks of paths are considered independent from each other.

3.1 Generation of Shortest Paths

The method for constructing a solution begins with the generation of paths. Therefore, q shortest paths between each origin-destination pair in the network are found, where q is a parameter given during implementation. Three different types of arc lengths were considered, and in order to assign the same importance to each type of length, q/3 different paths were obtained for each one of those different lengths:

$$1.\ lenght_{ij}^{k} = \left(C_{ijk} + f_{ij} \right) \left(1 + \left| \frac{\left(u_{ij} - D^{k} \right)}{D^{k}} \right| \right)$$

$$2.\ lenght_{ij}^{k} = \left(C_{ijk} + f_{ij} \right) \left(1 + \left| \frac{\left(u_{ij} - \sum_{h \in H} D^{h} \right)}{\sum_{h \in H} D^{h}} \right| \right)$$

$$3.\ lenght_{ij}^{k} = \left(C_{ijk} + f_{ij} \right) \left(1 + \frac{\phi_{ij}}{\phi'} \right)$$

where:

$$D^{k} = max_{s \in S} \left\{ d_{k}^{s} \right\} \qquad \forall\, k \in K$$

$$C_{ijk} = E\left(c_{ijk}^{s} \right) = \sum_{s \in S} \mathrm{p_s} \left(\mathrm{c_{ijk}^{s}} \right) \ \forall\, (i, j) \in A', k \in K \quad \textbf{(6)}$$

The list of q shortest paths for commodity k will be denoted by LP^k .

3.2 Constructing a solution

As it is the case in GRASP implementations, the constructive phase has two main features: adaptive greedy measures, and random selection. In addition, adaptive memory features have been included here, as proposed by Fleurent and Glover [5] to retain and analyze the characteristics of selected solutions and so providing a base for improving later executions of the constructive process. A set S_{sol} of r elite solutions is generated and updated (if necessary) each time a solution is generated. At first, S_{sol} contains r "null" solutions with infinite cost, where r is a parameter given during implementation. As long as feasible solutions are being generated, they may replace those in S_{sol} .

The constructive mechanism produces a solution incorporating one element (*i.e.* a *path*) at a time, so in each step of the process, there is at hand a partial solution. An element that can be selected as part of a partially constructed solution is called a *candidate element*. To determine which element will be selected to be included in a partial solution a greedy function is used. In order to introduce some randomness to this procedure a *restricted candidate list* (RCL) is used for each commodity. This list, from which an element is randomly selected, is formed by high quality elements, that is, candidate elements with the best values of their greedy function. The length of this list has been considered fixed.

Once an element has been added to the partially constructed solution, the values of the greedy function must be reevaluated. This makes the procedure acquire the *adaptivity* feature that characterizes it.

3.2.1 Evaluating the paths

The selection of a path must answer the following question: "which paths are most likely to be selected from our set of q shortest paths?"

In order to learn from previously generated solutions and previously selected paths, some sort of evaluation is needed. Let $value(\sigma^*, p)$ be the function which evaluates the benefit of including path p in σ^*, the solution being created. This function is defined in such a way that larger values correspond to better choices. Now, let's define the function $intensity(\sigma^*, p)$, larger values of it means that this choice occurs more frequently in the best members of S_{sol}.

An evaluation that takes into account both value and intensity of a choice may be defined as a monotone increasing function of its arguments as follows:

$$E(\sigma^*, p) = \lambda\, value(\sigma^*, p) + intensity(\sigma^*, p) \qquad \textbf{(7)}$$

Different values of λ in the will cause more emphasis on the diversity or on the intensity term which in turn will guide the selection of paths. Below we describe each of this functions.

To define the function *value* other measures must first be established. Let $C(B^k, p)$ be the function that represents the cost of assigning path p to B^k:

$$C(B^k, p) = \sum_{s \in S} p_s \left[\sum_{(i,j) \in path\ p} \left(c^s_{ijk}\, d'^s_k + f'_{ij} \right) \right] + \omega^0 \qquad \textbf{(8)}$$

Where:

$$\omega^0 = \omega \sqrt{\frac{\sum_{s \in S^+} p_s \left(z^s_{ijk} - E\left(z^s_{ijk}\right)\right)^2}{\sum_{s \in S^+} p_s}}$$

and p is any path from LP^k; f'_{ij} equals f_{ij} if edge $\{i, j\}$ has not yet been used, or 0 if it already has. u'_{ij} is the residual capacity of edge $\{i, j\}$; d'^s_k is the residual demand for commodity k (that is, the portion of the demand for commodity k waiting to be transported) under scenario s, $z^s_{ijk} = c^s_{ijk} d'^s_k$ and $E\left(z^s_{ijk}\right) = \sum_{s \in S} p_s z^s_{ijk}$.

The cost of this partially constructed solution σ^* if path p is assigned to B^k can be obtained as follows:

$$T^k(\sigma^*, p) = \sum_{m=1}^{k-1} \sum_{l=1}^{b^m} C(B_m, l) + \sum_{l=1}^{p} C(B_k, l) \qquad \textbf{(9)}$$

where G^m is the set of paths contained in each B^m. Define now $best\_cost(LP^k) = \min\{T(\sigma^*, l) : l \in LP^k\}$ as the minimal cost in that list. The cost of each path is normalized considering $best\_cost(LP^k)$ in such a way that the value of the function defined for each p belonging to that list is calculated as:

$$value(\sigma^*, p) = best\_cost(LP^k) / T(\sigma^*, p) \qquad \textbf{(10)}$$

It can be noticed that function $value(\sigma^*, p)$ will take values in the range [0,1].

Consider now the set S_{sol} of elite solutions. Let $Cost(\sigma)$ be the cost of a solution measured by the value of its objective function. Define $best\_cost(S_{sol}) = \min\{Cost(\sigma) : \sigma \in S_{sol}\}$. The cost of each solution in set S_{sol} is then normalized in the same way that was done for paths. Therefore, a function *Value* is defined for each solution in S_{sol} as follows:

$$Value(\sigma) = best\_cost(S_{sol}) / Cost(\sigma) \qquad \textbf{(11)}$$

This value represents a crude measure of the "strength" of paths in σ. Now, define the intensity function of assigning path p to the new solution σ^*. This function will take into account the frequency of this path in set S_{sol}.

$$intensity(\sigma^*, p) = \sum_{\{\sigma \in S_{sol} \mid p \in B^k\}} Value(\sigma) \qquad \textbf{(12)}$$

Paths in LP^k are ordered according to their evaluation function (7) **.**

3.2.2 Construction of RCL^k and Selection of Paths from LP^k

A fixed length for the RCL^k is considered and only those best evaluated paths, according to function (7) are included. Experiments showed that a 10% of q is appropiate.

In addition to what has been said, and in order to give the best evaluated paths a greater opportunity of being selected, $E(s^*, p)$ will be mapped into positive values over RCL^k. Then the probability of selecting path p from RCL^k, denoted $\Pi(s^*, p)$, is established to be strongly biased toward choosing the members of RCL^k with larger $E(s^*, p)$ values.

For each path in RCL^k, the probability of being selected is defined as follows:

$$\Pi(s^*, p) = \frac{E(s^*, p)}{\sum_{l \in RCL^k} E(s^*, l)} \qquad \textbf{(13)}$$

3.2.3 Updating Set of Elite Solutions

The constructive process will be guided by using the set S_{sol} of elite solutions. Initially, S_{sol} contains r "null" solutions with infinite cost, where "r" is a parameter given during the implementation phase. First "r" generated solutions by GRASP take place in S_{sol}. From that moment on, any solution being created will replace some other in S_{sol} if the following is achieved: it is better than the best solution in S_{sol} or else, it is better than the worst solution in S_{sol} but distant enough from the rest of solutions in S_{sol}.

The distance between two solutions considers how many different paths are in each of the solutions and is defined as follows:

$$\delta(s^1, s^2) = \frac{\sum_{k \in K} \sum_{p_j \in LP^k} \left| t^1_j - t^2_j \right|}{h_1 + h_2}$$

where $\mathbf{t}^i$ is the characteristic vector of solution s^i corresponding to the paths (taken from LP^k) included in the block of paths for each commodity k. h_i is the number of paths used in solution i.

3.3 Improving a solution

Once a block of paths is completed for a new solution being created, a routine called "improvement routine" is then executed. Basically, the improvement routine consists on sorting the paths to obtain a better distribution with lower cost, if this were possible.

4 Computational Experiment

C was used as the programming language to implement the procedures, which were run on a SUN™ Ultra 10 computer with Solarix™ r.7 operating system. The optimal solutions of each scenario were obtained using Cplex 8.0 .

A total of 20 instances with 20 nodes, 140 edges (280 arcs), 10 commodities and 10 scenarios were generated to carry out the experiment. The following parameters were used:

d_k^s Uniformly distributed in the range [60,100].

c_{ijk}^s Uniformly distributed in the range [60, 100].

u_{ij} Uniformly calculated in the following range:

$$\left[0.8 \cdot D^k, 1.2 \cdot D^k\right]$$

A ratio r is calculated for each problem. This ratio takes into account the fixed and variable costs and is used to classify the generated instances according to the relative importance of the former over the latter.

$$fixed\ cost\ ratio^s = |K| \frac{\sum_{(i,j)\in A} f_{ij}}{\sum_{k\in K} d_k^s \sum_{(i,j)\in A} c_{ijk}^s}$$

For instances with predominant fixed costs (f) this ratio varies between 1 and 1.5 ; for instances with predominant variable costs (v) varies between 10 and 12.

Loosely and tightly capacitated instances were obtained applying the following capacity ratio:

$$capacity\ ratio^s = |A| \frac{\sum_{k\in K} d_k^s}{\sum_{(i,j)\in A} u_{ij}}$$

and the ranges considered were: (6-8) for loose instances(l) and (12-16) for tight instances (t).

For each one of the instances the procedure was tested and the network design obtained is evaluated in each scenario and compared to the optimal solution in this scenario.

The first column shows the name of each instance. Column 2 and 3 show the expected cost, across all scenarios, of the optimal solutions found by Cplex, and of the solutions found by our procedure respectively. The percentages shown in column 4 are the expected deviations across all scenarios.

	CPLEX 1	GRASP	GRASP
Instance	Expected Cost	Expected Cost	Vs. CPLEX 1
201010fl01	939,726.26	940,174.13	0.05%
201010fl02	1,256,226.46	1,398,482.88	11.32%
201010fl03	945,933.69	1,018,115.63	7.63%
201010fl04	1,090,628.81	1,156,137.88	6.01%
201010fl05	1,162,295.69	1,162,857.50	0.05%
Average "fl"			**5.01%**
201010vl01	201,019.00	217,889.50	8.39%
201010vl02	197,683.03	204,936.69	3.67%
201010vl03	172,621.19	173,479.80	0.50%
201010vl04	183,960.64	192,547.72	4.67%
201010vl05	176,044.25	177,474.84	0.81%
Average "vl"			**3.61%**
201010ft01	2,488,105.76	2,556,587.50	2.75%
201010ft02	2,404,639.78	2,622,363.00	9.05%
201010ft03	2,435,707.04	2,563,850.00	5.26%
201010ft04	2,203,944.35	2,403,081.00	9.04%
201010ft05	2,329,129.67	2,526,654.75	8.48%
Average"ft"			**6.92%**
201010vt01	316,599.11	333,445.44	5.32%
201010vt02	327,795.00	351,654.97	7.28%
201010vt03	341,061.99	355,966.50	4.37%
201010vt04	342,302.61	356,045.78	4.01%
201010vt05	314,085.75	356,282.22	13.43%
Average "vt"			**6.88%**
Total Average			**5.61%**

Table 1.

5 Conclusions

In this paper, the robust network design problem with finite capacities was addressed. Solving this kind of problems is a difficult, but an important task and more research in this area has to be undertaken.

Computational experiments have shown that the proposed procedure is effective when tackling a network design problem with considerable uncertainty in data. Size, number of commodities and tight capacities make this problem very difficult to solve. Results show that fixed costs are highly relevant when solving instances with this characteristic. When fixed costs are not very high, impact in results is not very relevant.

To the best of our knowledge, no other techniques have been used to solve this model, and the tested procedure represents a great advance. Nevertheless it is important to keep in mind that the solutions found by the procedure developed here are considered initial solutions , so suitable post-processing routines can be designed to improve them.

Acknowledgements
This work has been partially supported by Conacyt under grant 36669-A

References:

[1] Karim de Alba, Ada Álvarez , José Luis González Velarde, Grasp with Adaptive Memory Programming for Finding Good Starting Solutions to the Multicommodity Capacited Network Design Problem. In *Interfaces in Computer Science and Operations Research,* H. Bhargava and Nong Ye (Eds), Kluwer, 2001, pp. 121-138

[2] Balakrishnan A., Magnanti T.L., A dual ascent procedure for large-scale uncapacitated network design, *Operations Research*, Vol. 37, No. 5, 1989 pp. 716-740.

[3] CPLEX Optimization, Inc., Incline Village, NV. ILOG CPLEX 8.0, *Reference Manual*, 2002

[4] Crainic, T.G., Frangioni, A., and Gendron, B. Bundle-based relaxation metods for multicommodity capacitated fixed-charge network design problems. *Publication CRT-98-45, Centre de recherche sur le transports, Université de Montreal,* 1998.

[5] Fleurent C. and Glover F., Improved Constructive Multistart Strategies for the Quadratic Assignment Problem Using Adaptive Memory, *Journal on Computing,* Vol. 11 No. 2, 1999, pp.198-204.

[6] Gutierrez, G.J., Kouvelis P., Kurawala A.A., A Robutness Approach to Uncapacitated Network Design Problems, *European Journal of Operational Research*, Vol. 94, No.2, 1996, pp.362-376.

[7] Holmberg K, Yuan D, A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem, *Operations Research*, Vo9l. 48, 2000, pp. 461-481.

[8] Kouvelis P., Yu G, *Robust Discrete Optimization and its Applications*, Kluwer Academic Publishers, 1997.

[9] Kubo M., Kasugi H., The Probabilistic Network Design Problem, *Journal of Operations Research*, Society of Japan, Vol. 35, No. 3, 1992, pp. 256-271.

[10] Laguna, M., González-Velarde J.L., A Benders-based Heuristic for the Robust Capacitated Internacional Sourcing Problem, To appear in *IIE Transactions*.

[11] Laguna M. Applying Robust Optimization to Capacity Expansion of One Location in Telecommunications with Demand Uncertainty, *Management Science*, Vol. 44, no. 11, 1997, pp. S101-S110.

[12] Mulvey J.M., Vanderbei R.J., Zenios S.A., Robust Optimization of Large-Scale Systems, *Operations Research*, Vol. 43. 1995, pp. 264-281.

[13] Pitsoulis L.S. and Resende M., Greedy Randomized Adaptive Search Procedures, In *Handbook of Applied Optimization*, P. M. Pardalos and M. G. C. Resende (Eds.), Oxford University Press, 2002, pp. 168-182.