Progressive Hedging-Based Meta-Heuristics for Stochastic Network Design

Teodor Gabriel Crainic
Xiaorui Fu
Michel Gendreau
Walter Rei
Stein W. Wallace

January 2009

CIRRELT-2009-03

Progressive Hedging-Based Meta-Heuristics for Stochastic Network Design

Teodor Gabriel Crainic^{1,2}, Xiaorui Fu^{3}, Michel Gendreau^{1,4}, Walter Rei^{1,2}, Stein W. Wallace^{5}

1. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)
2. Department of Management and Technology, Université du Québec à Montréal, C.P. 8888, succursale Centre-ville, Montréal, Canada H3C 3P8
3. Agence de la santé et des services sociaux de Montréal, 400 boul. De Maisonneuve Ouest, bureau 300, Montréal, Canada H3A 1L4
4. Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-ville, Montréal, Canada H3C 3J7
5. Molde University College, P.O. Box 2110, NO-6402 Molde, Norway

Abstract. We consider the stochastic variant of the fixed-charge capacitated multicommodity network design (CMND) problem in which demands are stochastic. We propose a two-stage stochastic programming formulation where design decisions make up the first stage, while a series of recourse decisions are made in the second stage to distribute the commodities according to observed demands. The overall objective is to optimize the cost of the first-stage design decisions plus the total expected distribution cost incurred in the second stage. To solve this formulation, we propose a metaheuristic framework inspired by the progressive hedging algorithm of Rockafellar and Wets. Following this strategy, scenario decomposition is used to separate the stochastic problem following the possible outcomes, or scenarios, of the random event. Each scenario subproblem then becomes a deterministic CMND problem to be solved. Our progressive hedging strategy takes advantage of specialized methods to solve effectively deterministic CMND problems. We also propose and compare different strategies aimed at penalizing non-consensus amongst scenario subproblems to approximate the global design. These strategies are then embedded into a parallel solution method, which is numerically qualified on a set of problem instances that also provide the means to examine the effects of demand correlations on the behavior of the proposed algorithm.

Keywords. Progressive hedging, Lagrangean relaxation, stochastic network design problems.

Acknowledgements. Partial funding for the project has been provided by Natural Sciences and Engineering Research Council of Canada (NSERC) through its Industrial Research Chair and Discovery Grant programs, by the contributions of the partners of the Chair, the Canada Foundation for Innovation, through its Leaders Opportunity Fund, the ministère de l'Éducation, du Loisir et Sport du Québec, SUN Microsystems, ILOG, UQAM, and Grant 171007/V30 from the Research Council of Norway.

---

* Corresponding author: Teodor-Gabriel.Crainic@cirrelt.ca

This document is also published as Publication #1329 by the Department of Computer Science and Operations Research of the Université de Montréal.

Dépôt légal – Bibliothèque et Archives nationales du Québec,
Bibliothèque et Archives Canada, 2009

1 Introduction

Fixed-charge capacitated multicommodity network design (CMND) models have been used to address many important planning problems in a variety of applications, such as transportation, logistics, and telecommunications, see [8] and [9]. In general terms, the CMND problem can be defined as follows: one must design a network that will be used to distribute a given set of commodities from different points of origin to different points of destination. The design decisions consist of choosing which capacitated arcs to select ("open") for inclusion in the network, considering that a fixed cost is paid whenever an arc is used for the first time. Once the design decisions are made, the commodities can be distributed through the network to satisfy the demands for transportation while respecting the available capacity on the selected arcs. The objective of the CMND problem consists of finding a design, i.e., a collection of arcs to be opened, which minimizes the total cost of the system computed as the sum of the total fixed cost and the total distribution cost.

CMND problems have mainly been studied under the assumption that all necessary information is available at the time when the design decisions are made, i.e., using deterministic CMND models. Thus, for example, the demands for transportation are considered known when one is deciding which arcs are to open. This assumption is rarely observed within realistic applications, however. One is usually faced with the challenge of having to make decisions when only limited information is available on the context prevailing at the time in the future when the designed network is to be used.

Stochastic programming [1] has been developed as a tool to explicitly introduce stochastic variability within the parameters of an optimization problem. Following this approach, optimization problems are formulated as having different decision stages according to when the uncertain information becomes known. The basic two-stage stochastic model, as the name implies, is defined as having two decision stages [1]. First stage decisions are made before the realization (occurrence) of the random events that influence the value of a set of parameters within the model. Once these decisions are fixed and the random events take place, all parameters become known. Second stage decisions are then used to adapt the solution given by the first stage decisions to the observed realization of the random events. The second stage decisions thus define the *recourse* actions that are available for the problem studied. The overall objective of the two-stage formulation being to optimize the "cost" of the first stage decisions plus the expected "cost" of the recourse given the probability distributions of the random variables.

Computing the recourse function, i.e., the expected cost of the recourse actions, is generally a daunting task. An often used approach is to approximate the general stochastic problem through a discretization of the probability distributions and the generation of a set of scenarios, each scenario representing a possible realization of the random event. By modeling uncertainty through scenarios for the CMND problem, the stochastic problem becomes a deterministic mixed integer linear program (MIP) of generally very large dimensions. Many techniques, such as the sample average approximation algorithm [6], use sampling iteratively to solve stochastic problems. To successfully apply such techniques to network design problems, one must be able to solve efficiently the approximated problems obtained [11]. The problem of identifying a set of representative scenarios will not be covered here. We suppose that such a set is provided and our aim is to design an efficient algorithm to solve the approximated, through the use of scenarios, stochastic CMND problem.

All parameters of CMND problems may display a stochastic nature in one application or another. One which is observed in almost all contexts, is the *demand* for transportation. This is the scope of this paper: the CMND problem with stochastic demands, addressed through a two-stage formulation, design decisions making up the first stage, while a series of recourse decisions are made in the second stage to distribute the commodities according to observed demands. Distribution costs are now dependent on the latter and the objective becomes to find a design that minimizes the sum of the total fixed cost and the total expected distribution cost (the recourse function).

Deterministic CMND problems belong to the NP-hard complexity class [8] and are thus difficult to solve in all but trivial cases. The difficulty in solving this type of problem is compounded when multiple scenarios are used to represent the transportation demand. One could simply note that the size of the associated MIP is generally beyond the reach of exact solution methods and that heuristic-based methodologies are required. As is the case with many stochastic programming models, since one is searching for a solution that is optimal, on average, one must plan for additional resources. Therefore, in general terms, when considering stochastic CMND problems, this means that a different, most likely larger, set of arcs has to be opened in order to hedge against the variability in the transportation demands. Finding which set of arcs to open, given the trade-offs that exist between the fixed costs incurred and the added flexibility and robustness for reducing the distribution costs and protecting against significant variations in demand, respectively, is at the heart of the complexity for this type of optimization problem.

In this paper, we use a metaheuristic framework inspired by the progressive hedging (PH) algorithm of Rockafellar and Wets [10] to address efficiently stochastic CMND problems. Following this strategy, scenario decomposition (SD) is used to separate the stochastic problem following the possible outcomes, or scenarios, of the random event. Each scenario subproblem then becomes a deterministic CMND problem to be solved. At this point, two main issues must be addressed: how should each scenario subproblem be solved? and how to use globally the local information yielded by the subproblems, particularly when scenarios do not agree on arc status, to guide the overall search mechanisms toward a unique design vector? Rockafellar and Wets [10] proposed an augmented Lagrangean strategy that converges to a global optimum in the case of continuous stochastic problems. At each iteration, an estimation of the solution is computed as the expectation over the current solutions of the scenario subproblems. The latter are then re-solved with adjusted penalties on the differences between the global estimation and the local solution. Unfortunately, the approach proposed in [10] may not converge in the integer case, and we therefore study a number of alternatives.

The contribution of this paper is twofold. First, we develop a metaheuristic framework based on the PH strategy that takes advantage of specialized methods to solve deterministic CMND problems. This type of general algorithmic framework was proposed in [7] and was successfully applied to the problem of stochastic lot-sizing [3]. To our knowledge, this paper proposes the first such approach for the case of stochastic CMND problems. Second, we propose and compare different strategies aimed at penalizing non-consensus amongst scenario subproblems to approximate the global design. These strategies are then embedded into a parallel solution method, which is numerically qualified on a set of problem instances that also provide the means to examine the effects of demand correlations on the behavior of the proposed algorithm.

This paper is organized as follows. Section 2 presents the stochastic CMND formulation addressed. We describe how SD is applied to the case of stochastic CMND problems in Section 3, while Section 4 introduces the different strategies that are proposed to obtain consensus amongst the scenario subproblems. The particular implementation of this methodology we use in this paper is presented in Section 5. Section 6 is dedicated to the presentation and analysis of the computational results. We conclude in Section 7.

2 The Stochastic CMND Model

In this section, we formulate the general stochastic CMND problem. Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with a node set $\mathcal{N}$ and a directed arc set $\mathcal{A}$. Assume $\mathcal{A}$ is the union of a set of design arcs $\mathcal{A}^{\text{A}}$ (for simplicity of presentation, we assume all arcs to be design arcs) and a set of dummy arcs $\mathcal{A}^{\text{D}}$ linking each pair of origin and destination nodes for which demand is defined. In many applications, in telecommunications, transportation and logistics, for example, these dummy arcs stand for the choice of using resources outside the system being planned. In a stochastic setting, they will also be part of the recourse strategy introduced later in this section.

Let $\mathcal{K}$ denote the set of commodities to be distributed using this network, and let the random vector $\mathbf{d}$ define the distribution demands within the problem. In the second stage, we consider that a number of random events $\omega \in \Omega$ may be observed. For a given realization ω, the distribution demands are fixed to $d(\omega)$ (i.e., $\mathbf{d} = d(\omega)$). Let i and j be the node indexes. The stochastic arc-based formulation of the CMND problem can then be written as follows:

$$\min \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \mathbf{E}_{\mathbf{d}}[Q(y, d(\omega))] \tag{1}$$

$$\text{s.t.} \quad y_{ij} \in \{0,1\}, \forall (i,j) \in \mathcal{A}, \tag{2}$$

where the binary variables y_{ij}, $\forall (i,j) \in \mathcal{A}$, represent the design decisions. Variable y_{ij} takes value one if arc $(i,j) \in \mathcal{A}$ is selected in the final design (to simplify the presentation, we assume $y_{ij} = 1$, $\forall (i,j) \in \mathcal{A}^{\text{D}}$). Otherwise, it is fixed to zero. Define value f_{ij}, for $(i,j) \in \mathcal{A}$, as the fixed cost that is incurred if arc (i,j) is opened, with $f_{ij} = 0$, $\forall (i,j) \in \mathcal{A}^{\text{D}}$. Then, (1) consists of minimizing the sum of both the total fixed costs (i.e., $\sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij}$) and the average distribution costs (i.e., $\mathbf{E}_{\mathbf{d}}[Q(y, d(\omega))]$, where function $Q(y, d(\omega))$ is set to equal the total distribution cost given design y and demands $d(\omega)$). Constraints (2) impose the integrality requirements of the design variables.

In problem (1)-(2), the function $Q(y, d(\omega))$, which is dependent on both the design decisions that are taken (i.e., y) and the demands that are observed (i.e., $d(\omega)$), represents the recourse cost that is incurred in the second stage. In order to properly formalize this function, let us first define how the random events influence the demands within the problem that is considered. In the case of deterministic CMND problems, transportation demands are expressed as follows: v^k units of commodity $k \in \mathcal{K}$ must be distributed from a single node of origin $o(k) \in \mathcal{N}$ to a single node of destination $s(k) \in \mathcal{N}$. Several types of uncertainty may be observed relative to this demand: volumes may vary, some forecast demands might not materialize, or an "unexpected" demand might pop up for an origin-destination not previously considered. In this paper, we focus on the first case, which is the most frequent and which also includes the second case when appropriate value ranges are considered. Then, if one considers that a random event $\omega \in \Omega$ influences the demand of a particular commodity $k \in \mathcal{K}$, for the problem under study, this influence involves the volume to be distributed (i.e., $v^k(\omega)$). Therefore, $\forall i \in \mathcal{N}$ and $\forall k \in \mathcal{K}$, for a given random event $\omega \in \Omega$, the demand value $d_i^k(\omega)$ is defined as follows:

$$d_i^k(\omega) = \begin{cases} v^k(\omega) & \text{if } i = o(k) \\ -v^k(\omega) & \text{if } i = s(k) \\ 0 & \text{otherwise.} \end{cases}$$

Once the first stage decisions are made (i.e., design y is fixed), and demands become known (i.e., vector $d(\omega)$ is fixed), in the second stage, one must decide how to distribute the commodities to satisfy all constraints. In general terms, the distribution decisions are based on a series of possible recourse strategies, which, in turn, define the recourse cost $Q(y, d(\omega))$. When developing a stochastic model, one of the more important concepts to formulate is the recourse strategies that are available given the problem that is studied. In the case of CMND problems, several recourse strategies may be considered, from the simple observe-and-pay-if-not-what-planned-for, to complex procedures modifying the design and the flow distribution over a number of stages. The choice is often determined by the application under study. In this paper, we consider a single-stage, network recourse expressed as flow decisions to be made, which combines 1) using the network designed in Stage 1 as well as possible, and 2) call upon extra capacity (from the dummy arcs) at a (unit) price. Such a strategy is adequate when first-stage design decisions cannot be readily altered such as, for example, transportation systems with published schedules and telecommunication systems. The function $Q(y, d(\omega))$ can then be formulated as follows:

$$
\begin{aligned}
Q(y, d(\omega)) = \quad \min \quad & \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} c_{ij}^k x_{ij}^k && (3) \\
\text{s.t.} \quad & \sum_{j \in \mathcal{N}^+(i)} x_{ij}^k - \sum_{j \in \mathcal{N}^-(i)} x_{ji}^k = d_i^k(\omega), \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K} && (4) \\
& \sum_{k \in \mathcal{K}} x_{ij}^k \leq u_{ij} y_{ij}, \quad \forall (i,j) \in \mathcal{A} && (5) \\
& x_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}. && (6)
\end{aligned}
$$

In (3)-(6), variable x_{ij}^k defines the amount of flow of commodity $k \in \mathcal{K}$ that is routed through arc $(i,j) \in \mathcal{A}$. Let value c_{ij}^k be the routing cost of one such unit of flow. Therefore, the objective (3) consists of minimizing the total distribution costs. Given vector $d(\omega)$, if $\mathcal{N}^+(i)$ and $\mathcal{N}^-(i)$ are respectively the sets of outward and inward neighbors of node i, then equations (4) define the network flow conservation constraints that require that the demands on all nodes be satisfied. As for equations (5), they impose the capacity restrictions on all arcs of the network. If arc $(i,j) \in \mathcal{A}$ is chosen in the design (i.e., $y_{ij} = 1$), then the total amount of flow that is routed through this arc cannot exceed capacity u_{ij}. Otherwise, if arc $(i,j) \in \mathcal{A}$ is not chosen (i.e., $y_{ij} = 0$), then capacity u_{ij} is unavailable. In this formulation, dummy arcs in $\mathcal{A}^{\text{D}}$ have no capacity restrictions. Finally, constraints (6) impose non-negativity on the flow variables. It should be noted that problem (3)-(6) represents a capacitated multicommodity minimum cost flow problem.

3 Applying SD to Stochastic CMND Problems

We begin this section by reformulating the general stochastic CMND problem. Let $\mathcal{S}$ define a set of possible scenarios for the random event (i.e., $\mathcal{S} \subseteq \Omega$). Using set $\mathcal{S}$, one can approximate the original stochastic CMND problem by using the following deterministic equivalent model:

$$
\begin{aligned}
\min \quad & \sum_{(i,j) \in \mathcal{A}} f_{ij} y_{ij} + \sum_{s \in \mathcal{S}} p_s \left(\sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} c_{ij}^k x_{ij}^{ks} \right) && && (7) \\
\text{s.t.} \quad & \sum_{j \in \mathcal{N}^+(i)} x_{ij}^{ks} - \sum_{j \in \mathcal{N}^-(i)} x_{ji}^{ks} = d_i^{ks} && \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S} && (8) \\
& \sum_{k \in \mathcal{K}} x_{ij}^{ks} \leq u_{ij} y_{ij} && \forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S} && (9) \\
& y_{ij} \in \{0, 1\} && \forall (i,j) \in \mathcal{A} && (10) \\
& x_{ij}^{ks} \geq 0 && \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S}, && (11)
\end{aligned}
$$

where p_s defines the probability associated with scenario $s \in \mathcal{S}$ and x_{ij}^{ks} represents the flow amount of commodity $k \in \mathcal{K}$ on arc $(i,j) \in \mathcal{A}$ if scenario $s \in \mathcal{S}$ is observed. Problem (7)-(11) is a large-scale mixed integer problem with a block-diagonal structure. Each block defined by constraints (8) and (9) represents a capacitated multicommodity minimum cost flow problem for $s \in \mathcal{S}$. Constraints (10) and (11) define the integrality and non-negativity restrictions on the decision variables. By solving problem (7)-(11), one finds a single design $y_{ij} \in \{0,1\}$, $\forall (i,j) \in \mathcal{A}$, that minimizes the sum of both the fixed costs and the average distribution costs overall scenarios included in $\mathcal{S}$.

One should note that constraints (9) link the first stage variables to the second stage variables. Considering the approximation of Ω provided by set $\mathcal{S}$, these constraints impose that $\forall s \in \mathcal{S}$, the flow distribution can only be made on the arcs that were opened in the first stage (i.e., $y_{ij} = 1$). Constraints (9) prevent the problem from being scenario separable. To apply the SD scheme, we first define the following vectors: $y^s_{ij} \in \{0,1\}$, $\forall (i,j) \in \mathcal{A}$ and $\forall s \in \mathcal{S}$. In doing so, a copy of the first stage variables is created for each scenario $s \in \mathcal{S}$. Model (7)-(11) can now be rewritten as follows:

$$\begin{aligned}
\min \quad & \sum_{s\in\mathcal{S}} p_s \left(\sum_{(i,j)\in\mathcal{A}} f_{ij} y^s_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} c^k_{ij} x^{ks}_{ij} \right) & & (12)\\
\text{s.t.} \quad & \sum_{j\in\mathcal{N}^+(i)} x^{ks}_{ij} - \sum_{j\in\mathcal{N}^-(i)} x^{ks}_{ji} = d^{ks}_i & \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S} & \quad (13)\\
& \sum_{k\in\mathcal{K}} x^{ks}_{ij} \le u_{ij} y^s_{ij} & \forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S} & \quad (14)\\
& y^s_{ij} = y^t_{ij} & \forall s,t \in \mathcal{S}, \quad s \neq t & \quad (15)\\
& y^s_{ij} \in \{0,1\} & \forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S} & \quad (16)\\
& x^{ks}_{ij} \ge 0 & \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S}. & \quad (17)
\end{aligned}$$

Equations (15) are referred to as the non-anticipativity constraints. These constraints are used to make sure that the design decisions are not tailored according to the scenarios considered in $\mathcal{S}$. All scenario designs (i.e., y^s_{ij}, $\forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S}$) must be equal to each other to produce what is referred to as a single implementable (see [10]) design. One can now make the important observation that if constraints (15) are relaxed, then problem (12)-(17) becomes scenario separable. However, it should be noted that the number of constraints defined in (15) may become quite large given the size of $\mathcal{S}$. Therefore, there are other ways of expressing the non-anticipativity constraints.

If $\overline{y}_{ij} \in \{0,1\}$, $\forall (i,j) \in \mathcal{A}$, is defined as the overall design vector (i.e., the design for all scenarios considered), then the following constraints are equivalent to (15):

$$\begin{aligned}
y^s_{ij} = \overline{y}_{ij} & \quad \forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S}, & (18)\\
\overline{y}_{ij} \in \{0,1\} & \quad \forall (i,j) \in \mathcal{A}. & (19)
\end{aligned}$$

Constraints (18) impose that each scenario design must be equal to the overall design (i.e., $\overline{y}_{ij}$, $\forall (i,j) \in \mathcal{A}$). As for (19), they are simply the required integrality conditions on the overall design. By using this particular formulation for the non-anticipative requirements, when Lagrangean relaxation is applied on (18), one can penalize individually the difference between the scenario solutions and the overall solution for each arc within the network. Therefore, the methodological approach proposed in this paper will be applied using (18)-(19).

Following the original decomposition scheme proposed in [10], constraints (18) are relaxed using an augmented Lagrangean strategy. We thus obtain the following objective for the overall problem:

$$\min \sum_{s\in\mathcal{S}} p^s \left(\sum_{(i,j)\in\mathcal{A}} f_{ij} y^s_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} c^k_{ij} x^{ks}_{ij} + \sum_{(i,j)\in\mathcal{A}} \lambda^s_{ij} (y^s_{ij} - \overline{y}_{ij}) + \frac{1}{2} \sum_{(i,j)\in\mathcal{A}} \rho (y^s_{ij} - \overline{y}_{ij})^2 \right), \quad (20)$$

where λ^s_{ij}, $\forall (i,j) \in \mathcal{A}$ and $\forall s \in \mathcal{S}$, define the Lagrangean multipliers for the relaxed constraints and ρ is a penalty ratio. Within function (20), let us consider the quadratic term: $\sum_{(i,j)\in\mathcal{A}} \rho (y^s_{ij} - \overline{y}_{ij})^2$.

When calculated, this term becomes: $\sum_{(i,j)\in\mathcal{A}} \rho(y_{ij}^s)^2 - 2\rho\overline{y}_{ij}y_{ij}^s + \rho(\overline{y}_{ij})^2$, which can be expressed as: $\sum_{(i,j)\in\mathcal{A}} \rho y_{ij}^s - 2\rho\overline{y}_{ij}y_{ij}^s + \rho\overline{y}_{ij}$, given the binary requirements for the design variables. Therefore, the objective of the relaxed problem can be formulated as follows:

$$\min \sum_{s\in\mathcal{S}} p^s \left(\sum_{(i,j)\in\mathcal{A}} \big(f_{ij} + \lambda_{ij}^s - \rho\overline{y}_{ij} + \frac{\rho}{2}\big) y_{ij}^s + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^{ks} \right) - \sum_{(i,j)\in\mathcal{A}} \lambda_{ij}^s \overline{y}_{ij} + \sum_{(i,j)\in\mathcal{A}} \frac{1}{2}\rho\overline{y}_{ij}. \quad (21)$$

Given the constraints of the model and considering the objective function (21), the relaxed problem is not scenario separable. However, if the overall design $\overline{y}_{ij}$, $\forall(i,j)\in\mathcal{A}$, is fixed to a given value vector (i.e., $[0,1]^{|\mathcal{A}|}$), then the model decomposes according to the scenarios included in set $\mathcal{S}$. All scenarios subproblems (i.e., $\forall s\in\mathcal{S}$) can then be expressed as follows:

$$\min \sum_{(i,j)\in\mathcal{A}} \big(f_{ij} + \lambda_{ij}^s - \rho\overline{y}_{ij} + \tfrac{\rho}{2}\big) y_{ij}^s + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^{ks} \quad (22)$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{N}^+(i)} x_{ij}^{ks} - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^{ks} = d_i^{ks} \qquad \forall i\in\mathcal{N}, \forall k\in\mathcal{K}, \forall s\in\mathcal{S} \quad (23)$$

$$\sum_{k\in\mathcal{K}} x_{ij}^{ks} \le u_{ij} y_{ij}^s \qquad \forall (i,j)\in\mathcal{A}, \forall s\in\mathcal{S} \quad (24)$$

$$y_{ij}^s \in \{0,1\} \qquad \forall (i,j)\in\mathcal{A}, \forall s\in\mathcal{S} \quad (25)$$

$$x_{ij}^{ks} \ge 0 \qquad \forall (i,j)\in\mathcal{A}, \forall k\in\mathcal{K}, \forall s\in\mathcal{S}. \quad (26)$$

For all scenarios $s\in\mathcal{S}$, within the subproblems (22)-(26), the Lagrangean multipliers λ_{ij}^s, $\forall s\in\mathcal{S}$, and the value ρ, are used to penalize, for all arcs $(i,j)\in\mathcal{A}$, the differences that may exist between the scenario designs and the fixed overall design, which serves as a reference point. Therefore, these penalties can be adjusted in order to drive all scenario subproblems to converge to a single design that is defined by $\overline{y}_{ij}$, $\forall(i,j)\in\mathcal{A}$. An important observation that must be made here is that when adjustments are made on the penalties, the subproblems (22)-(26) are reduced to deterministic CMND problems with modified fixed costs. From a methodological perspective, this turns out to be very interesting since one is now able to use some of the more efficient algorithms (either heuristic or exact) that have been developed for the deterministic case of the problem under study. However, how one is able to construct an overall design $\overline{y}_{ij}$, $\forall(i,j)\in\mathcal{A}$, to be used as reference remains to be determined. Furthermore, in general terms, the objective functions (22) suggest that by adjusting the fixed costs of the design variables within subproblems (22)-(26), one can search for consensus amongst all scenario designs. How should these adjustments be made is also a question yet to be answered.

4 Obtaining Consensus amongst Scenario Subproblems

In order to produce a general solution strategy for stochastic CMND problems using the decomposition scheme presented within the previous section, one must now develop iterative strategies to both define the general design that is used to separate the large scale problem and to fix the penalties when non-consensus is observed amongst the scenario subproblems. The different such strategies that are used within the solution approach that is proposed in this paper will now be presented. This section is divided in two subsections. In the first subsection, we present how the overall design is defined iteratively. As for the second subsection, it is used to present the different strategies for penalty adjustments.

4.1 Defining the overall design

To solve problem (7)-(11) using the SD scheme previously described, one must be able to utilize the local information provided by the scenario design solutions to produce a general design to be used over all scenarios. Let ν define an index count on the number of iterations made by a general algorithm that sequentially solves subproblems (22)-(26) $\forall s \in \mathcal{S}$ and then, produces a general design $\overline{y}_{ij}^{\nu}$, $\forall (i,j) \in \mathcal{A}$ using an aggregation operator on $y_{ij}^{s\nu}$, $\forall s \in \mathcal{S}$ and $\forall (i,j) \in \mathcal{A}$ (i.e., the solutions obtained for each scenario subproblem). The aggregation operator is simply defined as the general function that is used to combine the scenario solutions into a single general solution given a weight (or importance) for each scenario in set $\mathcal{S}$. Originally in [10], the average function was used as the aggregation operator. Therefore, in this case, the weights considered were simply the probabilities associated with the scenarios. When this idea is transposed to the present problem, one obtains the following aggregation operator:

$$\overline{y}_{ij}^{\nu} = \sum_{s \in \mathcal{S}} p_s y_{ij}^{s\nu}, \quad \forall (i,j) \in \mathcal{A}. \tag{27}$$

It should be noted that (27) does not necessarily produce a general feasible design. For a given arc $(i,j) \in \mathcal{A}$, considering all solutions $y_{ij}^{s\nu}$, $\forall s \in \mathcal{S}$, if one has consensus for all scenarios then one obtains $\overline{y}_{ij}^{\nu} \in \{0,1\}$. Otherwise, there is non-consensus and one observes $0 < \overline{y}_{ij}^{\nu} < 1$, which is infeasible given the integrality requirements on the design variables. It was observed in [10] that, by using the average function as the aggregation operator in the case of non-convex problems, the overall solution strategy may not converge to an optimal solution. Therefore, by simply using (27) in the case of stochastic CMND problems, one cannot guarantee that a good (or even feasible) solution will be obtained. However, this aggregation operator may still be used to guide the overall solution process. When there is non-consensus for a given arc $(i,j) \in \mathcal{A}$, the value $\overline{y}_{ij}^{\nu}$ provides information concerning the general trend amongst the scenario designs. If the value $\overline{y}_{ij}^{\nu}$ is low (i.e., close to zero), then there is incentive in closing arc (i,j) within the general design. Otherwise, if value $\overline{y}_{ij}^{\nu}$ is high (i.e., close to one), then the incentive is in opening arc (i,j) in the general design. Therefore, (27) will be used as a reference point within a first search phase with the objective of identifying the subset of arcs for which consensus is possible. A second search phase will then be applied on the restricted problem obtained by fixing all consensus arcs within the general design.

To produce a feasible solution using values $\overline{y}_{ij}^{\nu}$, $\forall (i,j) \in \mathcal{A}$, one can always apply a worst-case analysis and find what we refer to as the *max design*:

$$y_{ij}^{M\nu} = \bigvee_{s \in \mathcal{S}} y_{ij}^{s\nu}, \quad \forall (i,j) \in \mathcal{A}. \tag{28}$$

Within the *max design*, all arcs that are opened in some scenario design are included in the overall design. This guarantees that the values $y_{ij}^{M\nu}$, $\forall (i,j) \in \mathcal{A}$, define a feasible design for problem (7)-(11). Therefore, $\sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij}^{M\nu} + \sum_{s \in \mathcal{S}} p_s Q(y^{M\nu}, d_s)$, where vectors $y^{M\nu}$ and d_s include respectively the values (28) and the demands for scenario $s \in \mathcal{S}$, defines an upper bound on the optimal value of problem (7)-(11). If the *max design* is calculated iteratively, then a best upper bound can be kept throughout the first search phase. It should be noted that although the *max design* can be used as the reference point within the SD scheme, whenever consensus is low within the scenario designs, (28) may overestimate the number of arcs that have to be opened within the overall design. Since the fixed costs are then adjusted to penalize non-consensus with respect to the reference point, this strategy may create a bias in the search process towards producing unnecessarily large overall designs. Therefore, solutions (28) are only used to produce the upper bound.

4.2 Strategies for penalty adjustments

As was previously observed, once the SD scheme is applied to problem (7)-(11), one is left with the task of having to solve a deterministic CMND problem for each scenario $s \in \mathcal{S}$. Given the scenario designs obtained and using the aggregation operator, as defined in the previous subsection, one generates a reference point that serves as the global design. To induce consensus amongst the scenario subproblems, the fixed costs are then adjusted within each subproblem to penalize non-concordance between the scenario designs obtained and the reference point generated. In this subsection, we present the two different strategies that are proposed to perform these adjustments.

4.2.1 Progressive hedging

The first strategy is the one originally proposed in [10]. Let us recall that within the SD scheme presented, an augmented Lagrangean relaxation is applied to the non-anticipativity constraints. Considering the algorithm that is performed in the first phase, for a given iteration ν, let $\lambda_{ij}^{s\nu}$ define the value of the Lagrangean multiplier associated with the relaxed non-anticipativity constraint for the design decision on arc $(i,j) \in \mathcal{A}$ for scenario $s \in \mathcal{S}$ and let ρ^{ν} define the value of the ratio for the quadratic penalty. When the strategy proposed in [10] is applied to the case of stochastic CMND problems, then the values $\lambda_{ij}^{s\nu}$ and ρ^{ν} are updated as follows, $\forall (i,j) \in \mathcal{A}$ and $\forall s \in \mathcal{S}$:

$$\lambda_{ij}^{s\nu} \leftarrow \lambda_{ij}^{s\nu-1} + \rho^{\nu-1}(y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu-1}) \tag{29}$$

$$\rho^{\nu} \leftarrow \alpha\rho^{\nu-1} \tag{30}$$

where $\alpha > 1$ is a given constant, ρ^0 is fixed to a positive value to ensure that $\rho^{\nu} \to \infty$ as the number of iterations ν increases, $y_{ij}^{s\nu}$ is the current value of the design decision associated with arc (i,j) and $\overline{y}_{ij}^{\nu-1}$ is the obtained value for arc (i,j) within the previous reference point.

Considering (29), for a particular $s \in \mathcal{S}$, there are three possible adjustments. If $y_{ij}^{s\nu} < \overline{y}_{ij}^{\nu-1}$, then the fixed cost associated with arc $(i,j) \in \mathcal{A}$ is reduced in the objective function of the scenario subproblem (i.e., $\lambda_{ij}^{s\nu} < 0$). In this case, the arc is closed in the scenario design (i.e., $y_{ij}^{s\nu} = 0$) but there was previously non-consensus (i.e., $0 < \overline{y}_{ij}^{\nu-1} < 1$). Therefore, the fixed cost is adjusted so as to give an incentive in opening the arc within the scenario design. If $y_{ij}^{s\nu} > \overline{y}_{ij}^{\nu-1}$, then the fixed cost associated with arc $(i,j) \in \mathcal{A}$ is augmented in the objective function of the scenario subproblem (i.e., $\lambda_{ij}^{s\nu} > 0$). In this case, the arc is opened in the scenario design (i.e., $y_{ij}^{s\nu} = 1$) and, once again, there was previously non-consensus (i.e., $0 < \overline{y}_{ij}^{\nu-1} < 1$). Therefore, the fixed cost is adjusted so as to give an incentive in closing the arc within the scenario design. Finally, if $y_{ij}^{s\nu} = \overline{y}_{ij}^{\nu-1}$, then there is consensus amongst the scenario designs and the fixed cost remains unchanged. As for update (30), it simply states that the value of the penalty ratio is steadily increased as the number of iteration grows.

Let $f_{ij}^{s\nu}$ refer to the adjusted fixed cost of arc $(i,j) \in \mathcal{A}$ in the objective of the subproblem associated with scenario $s \in \mathcal{S}$ at iteration ν. We then obtain Algorithm 1, by casting (29) and (30) within the iterative algorithm used for the first search phase, Within Algorithm 1, the scenario subproblems are solved iteratively with the fixed costs being updated using values (29) and (30). To apply the procedure, one still has to specify how each deterministic CMND problem (i.e., the scenario subproblems) is solved. Furthermore, stopping criterion have also to be defined. These questions are addressed in Section 5.

Algorithm 1 First phase using progressive hedging

```
Initialization
    $\nu \leftarrow 0$
    $\lambda_{ij}^{s\nu} \leftarrow 0 \ \forall (i,j) \in A, \forall s \in \mathcal{S}$
    $\rho^{\nu} \leftarrow \rho^{0}$
    for all $s \in \mathcal{S}$ do
        $f_{ij}^{s\nu} \leftarrow f_{ij}, \ \forall (i,j) \in \mathcal{A}$
        solve the corresponding CMND subproblem
    $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in \mathcal{S}} p^{s} y_{ij}^{s\nu} \ , \ \forall (i,j) \in \mathcal{A}$
    calculate and evaluate $y^{M\nu}$
    $bestSolution \leftarrow y^{M\nu}$
while stopping criterion are not met do
    $\nu \leftarrow \nu + 1$
    for all $s \in \mathcal{S}$ do
        $f_{ij}^{s\nu} \leftarrow f_{ij} + \lambda_{ij}^{s\nu-1} - \rho^{\nu-1}\overline{y}_{ij}^{\nu-1} + \frac{\rho^{\nu-1}}{2}, \forall (i,j) \in \mathcal{A}$
        solve the corresponding CMND subproblem
    Update
        $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in \mathcal{S}} p^{s} y_{ij}^{s\nu}$
        $\lambda_{ij}^{s\nu} \leftarrow \lambda_{ij}^{s\nu-1} + \rho^{\nu-1}(y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu-1})$
        $\rho^{\nu} \leftarrow \alpha\rho^{\nu-1}$
        calculate and evaluate $y^{M\nu}$
        update $bestSolution$ if $y^{M\nu}$ gives current best
```

4.2.2 Heuristic strategy

Since one has to solve a series of deterministic CMND problems with modified fixed costs when SD is applied to problem (7)-(11), it appears interesting to see how other strategies may be used to penalize non-consensus. In this section, we present a new heuristic penalty adjustment strategy that quickly targets the arcs for which the level of consensus amongst the scenario designs is high. At the end of iteration ν, one obtains values $\overline{y}^{\nu}_{ij}$, which, as was observed earlier, can be viewed as defining a trend amongst the scenario designs to either open or close arc (i, j). A lower value of $\overline{y}^{\nu}_{ij}$ translates the fact that arc (i, j) is opened only in a small portion of the scenario designs, while a higher value means that arc (i, j) is opened within a majority of scenario designs. Therefore, one can argue that if value $\overline{y}^{\nu}_{ij}$ is less than a given threshold c^{low}, then increasing the fixed cost of arc (i, j) may drive the subproblems to avoid using it. On the other hand, if value $\overline{y}^{\nu}_{ij}$ is more than a given threshold c^{high}, in order to attract the subproblems to use arc (i, j), the fixed cost should be lowered. Therefore, one can define the following adjustment strategy:

$$f^{\nu}_{ij} = \begin{cases} \beta f^{\nu-1}_{ij} & \text{if } \overline{y}^{\nu-1}_{ij} < c^{low} \\ \frac{1}{\beta} f^{\nu-1}_{ij} & \text{if } \overline{y}^{\nu-1}_{ij} > c^{high} \\ f^{\nu-1}_{ij} & \text{otherwise,} \end{cases} \tag{31}$$

where β is a constant larger than 1, c^{low} and c^{high} are two constants such that $0 < c^{low} < 0.5$ and $0.5 < c^{high} < 1$, and f^{ν}_{ij} represents the modified fixed cost of arc (i, j) at iteration ν.

The above adjustment is referred to as being *global*, since the fixed cost modifications are made for all scenarios. Keeping in mind that the ultimate aim of the search procedure in the first phase is to obtain a unanimous design, this heuristic modification can be pushed even further by modifying the fixed costs locally within the scenario subproblems. For a given scenario $s \in \mathcal{S}$, if the difference between values $y^{s\nu}_{ij}$ and $\overline{y}^{\nu}_{ij}$ is large, then one may be interested in emphasizing the adjustment to the fixed cost within the objective function of this particular scenario subproblem. This modification is considered to be *local* in the sense that it only affects the subproblem of scenario s at the current iteration. One thus defines the following update:

$$f^{s\nu}_{ij} = \begin{cases} \beta f^{\nu}_{ij} & \text{if } |y^{s\nu-1}_{ij} - \overline{y}^{\nu-1}_{ij}| \geq c^{far} \text{ and } y^{s\nu-1}_{ij} = 1 \\ \frac{1}{\beta} f^{\nu}_{ij} & \text{if } |y^{s\nu-1}_{ij} - \overline{y}^{\nu-1}_{ij}| \geq c^{far} \text{ and } y^{s\nu-1}_{ij} = 0 \\ f^{\nu}_{ij} & \text{otherwise,} \end{cases} \tag{32}$$

where $0.5 < c^{far} < 1$ and $\beta > 1$ are two given constant parameters, and $f^{s\nu}_{ij}$ stands for the modified local fixed cost of arc (i, j) for scenario s at iteration ν. In (32), parameter c^{far} defines the threshold at which point a local adjustment to the fixed cost is applied.

Using a similar idea, one can apply a variable fixing strategy to reduce the size of the subproblems being solved. If for scenario s, one observes that $|y^{s\nu-1}_{ij} - \overline{y}^{\nu-1}_{ij}| \leq c^{near}$, then the decision on the arc status should be kept by fixing $y^{s\nu}_{ij}$ to $y^{s\nu-1}_{ij}$. In this case, if one has sufficient consensus, which is expressed using threshold $0 < c^{near} < 0.5$, on the status of arc (i, j), then the design decision for this arc is fixed locally. When both the adjustment strategies (31) and (32) as well as the variable fixing strategy are used within the iterative algorithm performed in the first search phase, one obtains Algorithm 2.

Algorithm 2 First phase using heuristic strategy

```
Initialization
    $\nu \leftarrow 0$
    for all $s \in \mathcal{S}$ do
        $f_{ij}^{s\nu} \leftarrow f_{ij}, \forall (i,j) \in \mathcal{A}$
        solve the corresponding CMND subproblem
    $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in \mathcal{S}} p^s y_{ij}^{s\nu}, \forall (i,j) \in \mathcal{A}$
    calculate and evaluate $y^{M\nu}$
    $bestSolution \leftarrow y^{M\nu}$
while stopping criteria are not met do
    $\nu \leftarrow \nu + 1$
    $\forall (i,j) \in A$, modify $f_{ij}^{\nu}$ globally using equation (31)
    for all $s \in \mathcal{S}$ do
        $\forall (i,j) \in \mathcal{A}$, modify $f_{ij}^{s\nu}$ locally using equation (32)
        fix some $y_{ij}^{s\nu}$ if needed
        solve the corresponding CMND subproblem
    Update
        $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in \mathcal{S}} p^s y_{ij}^{s\nu}, \forall (i,j) \in \mathcal{A}$
        calculate and evaluate $y^{M\nu}$
        update $bestSolution$ if $y^{M\nu}$ gives current best
```

5 Implementations

A number of algorithmic components need to be specified for a complete description of the methodology implementation used to perform the computational experiments reported in the next section, principally, how subproblems are addressed, how the algorithms are stopped, and the parallel implementation.

For each scenario and iteration, the resulting subproblem is a deterministic CMND with possibly modified fixed costs. To address these problems, we use one of the currently best procedures for the CMND, the Cycle-based tabu search of Ghamlouche, Crainic, and Gendreau [2], which explores the space of the arc-design variables by re-directing flow around cycles and closing and opening design arcs accordingly. The initial solution is simply obtained by opening all arcs, solving the related minimum cost network flow problem (capacitated, multicommodity), and closing the unused arcs. Later, at each iteration of the global algorithm, each subproblem is solved starting from the solution of the previous iteration.

As indicated earlier on, there are not, yet, theoretical criteria for the convergence of the PH algorithm in integer cases. Meta-heuristics usually stop on maximum limits on CPU time, number of iterations, number of consecutive iterations without improvement, and so on. Yet, letting the method simply stop on such criteria might result in a "solution" where consensus has not been obtained for all design arcs. We proceed therefore in two phases. The first corresponds to the algorithms introduced in the previous section, which stop either when (if) consensus is achieved for all design arcs, or, once one of the following criteria is satisfied: 50 iterations, 10 consecutive non-improving iterations, 10 hours of CPU time. The second phase then solves the restricted stochastic CNMD obtained by fixing all design arcs for which consensus has been achieved. In the current implementation, this problem is solved by branch-and-bound using CPLEX.

One of the major issues in addressing stochastic CNMD through scenario decomposition, as most other combinatorial optimization problems, is the size of the corresponding deterministic problem. Parallel computation can and has been used to address this issue and it is adopted here as well. In the current version, we use a simple master-slave synchronous strategy, where the master controls the search, computes the global design, and performs the parameter updates, while the slave processors modify the fixed costs according to the information received from the master and solve the resulting scenario subproblems. Synchronization is performed at each iteration, that is, once all scenarios have been addressed.

6 Numerical Results

In this section, we report the experimental results and analyze the performance of the different proposed algorithms. Two instance sets, denoted respectively **S** and **R**, are used to perform the experiments. There are 16 problems included in set **S**. These problems are derived from the instances used in [4] for the time-dependent stochastic service network design problem. In order to obtain instances for our problem setting, we first dropped the time periods from the problems for which the associated network is a complete graph. The vehicle capacity defined in [4] was then used as the capacity for all non-dummy arcs in the network. Since all arcs have the same capacity, a unique value was assigned to all fixed costs. For each commodity, only the origin and destination nodes were used within the stochastic CMND instances created. Both the available time period and the time period at which the commodity is required were simply ignored. An artificial arc between the origin and destination nodes was added for each commodity in the problem. Finally, concerning the size of the instances in set **S**, it should be noted that they are based on networks of 16 and 30 nodes, with 14, 40 and 80 commodities and 10, 20, 60 and 90 scenarios.

Table 1: Attributes of **R** instances

Name	$\|\mathcal{N}\|$	$\|\mathcal{A}\|$	$\|\mathcal{K}\|$
r04	10	60	10
r05	10	60	25
r06	10	60	50
r07	10	82	10
r08	10	83	25
r09	10	83	50
r10	20	120	40
r11	20	120	100

A second set of stochastic instances was obtained from the deterministic CMND problem set **R** used in [2]. Eight deterministic instances were selected from set **R** (r04 to r11 inclusively), along with five different combined levels of fixed cost and capacity ratios (1, 3, 5, 7 and 9). Attributes of these instances are described in Table 1. To generate stochastic versions, it was first assumed that the demands for the problems followed triangular distributions. To define such distributions, three parameter values need to be specified: min a, max b and mode c. For each commodity $k \in \mathcal{K}$, the mode c, which represents the most likely outcome, was fixed to be the original deterministic value for the volume associated with the demand (i.e., $c = v^k$). We then set the other two parameters: $a = 0$ and $b = 1.25c$. Demands were then assumed to be linearly correlated and three different levels of positive correlations were considered to create different instances. The scenario trees were generated using the procedure proposed in [5], and instances for which the total number of scenarios was set to 16, 32 and 64 were created. A total of 360 instances were obtained in this second set.

All programs are coded in C++. Numerical experiments were conducted on a Sun Fire X4100 cluster of 16 computers, each computer having two 2.6 GHz Dual-Core AMD Opteron processors and 8192 Megabytes of RAM, operating under Solaris 2.10. The parallel implementation used four slave processors and communications were implemented using OpenMP. Finally, it should be specified that CPLEX version 10.1.1 was used to solve the capacitated multicommodity network flow problems within the cycle-based tabu search and for the branch-and-bound applied to the complete multi-scenario deterministic problem and the restricted S-CMND on Phase II.

To obtain the numerical results presented in this section, the algorithms formalized in subsection 4.2 are first applied to the problem set **S** and compared to CPLEX. Given the complexity of solving the complete multi-scenario deterministic problems in set **S**, a maximum time of 600 minutes CPU time (with the exception of one instance for which the maximum time was doubled) was imposed on the solution process of CPLEX, which serves here as the benchmark for all comparisons. Overall, these first results are used to analyze the general performance of the proposed strategies. In a second set of tests, all algorithms are used to solve the instances of set **R**. Once again, CPLEX is applied on the complete multi-scenario deterministic problems with a maximum alloted time of 500 minutes CPU time. The results obtained on set **R** are used to better compare the different proposed algorithms. In particular, we analyze the impact on the solutions obtained by the heuristics when local adjustments (32) are made within the fixed costs. Furthermore, we look at how results vary according to both the correlation level of the demands and with the combined levels of fixed cost and capacity ratios.

Concerning the implementations that are tested, it should be noted that Tabu-PH and Tabu-HC refer to the meta-heuristics that use the cycle-based tabu search procedure of [2] to solve the CMND subproblems and that apply, respectively, the progressive hedging and heuristic strategies to adjust the penalties. As for the parameter values, in the case of Tabu-PH, α is set to 1.1 and ρ^0 is set to $1 + log(1 + D^0)$, where D^0 is the inconsistency level (i.e., the number of arcs for which there is non-consensus amongst the scenario solutions) after the initialization phase. For Tabu-HC, both global and local fixed cost adjustments are performed with a penalizing ratio of $\beta = 1.1$ and thresholds are set to $c^{high} = 0.8, c^{low} = 0.2$ for global adjustments, and $c^{far} = 0.7, c^{near} = 0.2$ for local adjustments. Finally, it should be noted that all parameters defined for the cycle-based tabu search procedure are set to the values originally used in [2].

Table 2: Results on set **S**: CPLEX vs. Hedging strategies

Problems			CPLEX				Tabu-PH			Tabu-HC		
$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	L.B.	U.B.	Gap	Time	Val.	Gap	Time	Val.	Gap	Time
16	14	10	4909.3	4909.3	0.00%	0.19	4909.3	0.00%	0.34	4909.3	0.00%	0.34
16	14	20	4990.1	4990.1	0.00%	0.79	4990.1	0.00%	0.66	4990.1	0.00%	0.65
30	14	10	5198.6	5198.6	0.00%	0.18	5198.6	0.00%	2.69	5198.6	0.00%	2.58
30	14	20	5218.6	5218.6	0.00%	0.43	5218.6	0.00%	5.96	5218.6	0.00%	5.88
16	40	20	15184.9	15184.9	0.00%	76.16	15243.1	0.38%	4.07	15243.1	0.38%	3.79
16	40	60	15112.8	15244.7	0.87%	600.11	15196.3	0.55%	11.45	15196.3	0.55%	12.40
16	40	90	15103.9	15204.8	0.67%	600.32	15194.3	0.60%	18.53	15194.3	0.60%	20.28
30	40	20	14056.5	14301.0	1.74%	600.17	14498.9	3.15%	133.65	14321.9	1.89%	132.61
30	40	60	13409.3	14723.1	9.80%	600.48	14350.1	7.02%	199.98	14317.9	6.78%	201.96
30	40	90	12787.0	14723.0	15.14%	600.81	14321.4	12.00%	230.03	14287.8	11.74%	232.61
16	80	20	26773.0	27167.5	1.47%	600.23	27464.4	2.58%	5.71	27359.9	2.19%	13.25
16	80	60	26330.8	28621.4	8.70%	600.51	27272.0	3.57%	216.30	27190.2	3.26%	513.33
16	80	90	25709.6	28621.1	11.32%	600.55	27347.4	6.37%	522.49	27371.2	6.46%	524.64
30	80	20	29303.9	31408.3	7.18%	600.46	31010.6	5.82%	217.06	30913.5	5.49%	217.28
30	80	60	27491.8	31412.7	14.26%	600.86	30874.2	12.30%	317.47	30829.7	12.14%	346.34
30	80	90	27473.0	31412.4	14.34%	1201.78	30704.5	11.76%	287.27	30627.4	11.48%	312.68
Average			16815.82	18021.34	5.34%	455.25	17737.11	4.13%	135.85	17698.11	3.94%	158.79

In Table 2 are reported the detailed results obtained by all algorithms when applied to problem set **S**. It should be specified that L.B., U.B., Gap, and Time values reported for CPLEX refer, respectively, to the best lower and upper bounds found by the procedure, the optimal gap obtained with these values, and the total computation time expressed in minutes. For Tabu-PH and Tabu-HC, Val., Gap and Time represent the value of the best solution obtained by the meta-heuristic, the optimal gap of this solution, which is calculated using the lower bound of CPLEX, and again the total computation time expressed in minutes, respectively.

When analyzing the results reported in Table 2, one first observes that the complexity associated with the solution of the complete multi-scenario deterministic problem becomes significantly higher as the number of commodities to be distributed through the network increases. The CPLEX procedure solves efficiently (i.e., in less than one minute of computation time) those instances for which $|\mathcal{K}| = 14$. When comparing these results with those obtained by Tabu-PH and Tabu-HC, one observes that the meta-heuristics both find the optimal solutions using slightly more computation time (i.e., up to six minutes of computation time). Although, the meta-heuristics are slower than CPLEX on these instances, it should noted that these remain easy instances that all algorithms are able to solve efficiently. When $|\mathcal{K}|$ increases to 40, the instances become harder to solve using CPLEX. When considering those instances for which $|\mathcal{N}| = 16$, CPLEX is able to solve only one in the maximum alloted time. On the two other instances, the procedure comes close but is unable to complete the search in 600 minutes of computation time. In contrast, one observes that both the Tabu-PH and Tabu-HC meta-heuristics find near optimal solutions (i.e., with a gap that is less than one percent), using a fraction of the time (the average CPU times of CPLEX, Tabu-PH, and Tabu-HC are 425.53, 11.35, and 12.16 minutes, respectively, for these instances).

For all other instances, CPLEX failed to solve the complete multi-scenario deterministic problem in the maximum time allowed and, in all but two cases, was outperformed by both Tabu-PH and Tabu-HC

Table 3: Results on set **R**: CPLEX vs. Hedging strategies (without local adjustments)

		CPLEX					Tabu-PH			Tabu-HC		
Corr.	$\|\mathcal{S}\|$	L.B.	U.B.	Sol.	Gap	Time	Val.	Gap	Time	Val.	Gap	Time
0	16	410810	458704	31	4.31%	157.47	424509	1.60%	67.53	424593	1.64%	63.59
	32	403121	471938	27	8.46%	194.85	455834	5.44%	139.21	455725	5.47%	135.76
	64	385601	484601	22	15.94%	256.35	464468	10.93%	186.98	462378	10.47%	186.79
0.2	16	414214	459218	30	4.10%	159.94	428347	1.59%	62.28	428429	1.62%	52.41
	32	406689	473296	26	7.89%	196.47	453816	4.80%	139.15	453655	4.80%	132.99
	64	387049	488098	24	16.10%	249.54	462564	10.54%	187.81	460963	10.25%	188.34
0.8	16	408172	458136	31	4.62%	144.81	423953	1.75%	67.85	423743	1.72%	66.94
	32	407050	476565	27	8.31%	195.38	459702	5.39%	141.31	459061	5.36%	130.43
	64	391023	487397	22	14.38%	270.52	469979	10.11%	193.65	469038	10.03%	194.71

in less computation time. Considering these instances, on average, CPLEX obtained a solution gap of 9.33% in 667.32 minutes CPU time, compared to the average solution gaps of Tabu-PH and Tabu-HC which were, respectively, of 7.18% and 6.83%, for average CPU times of 236.66 and 277.19 minutes. These results seem to show that both meta-heuristics are more efficient addressing the instances of problem set **S** when compared to a direct solution approach using CPLEX. As to which of the two strategies works best, one observes that while Tabu-HC obtains, on average over all instances, the better results (i.e., a gap of 3.94% compared to 4.13% for Tabu-PH), it does so using, on average, more computation time (i.e., 158.79 minutes compared to 135.85 minutes for Tabu-PH).

Tables 3 and 4 display aggregated results on problem set **R** for the hedging strategy variants without and with local adjustments, respectively. Most column headings for these tables have a meaning similar to those in Table 2. In both tables, results are averaged out for the 40 **R** instances generated for each of the three correlation levels (Corr.) and each of the three sizes of the set of scenarios ($|\mathcal{S}|$). Column "Sol." indicates how many of the 40 instances considered were solved to optimality by CPLEX within 500 minutes of CPU time.

A first conclusion that can be drawn from these tables is that the inclusion of local adjustments in the hedging strategies seems to have very little impact both on the quality of the solutions obtained and on the CPU times observed. This impression is confirmed by a comparison of results obtained on individual instances (available from the authors), which reveals that both variants always perform very similarly. The remainder of our analysis will thus focus on the variant without local adjustments.

Table 4: Results on set **R**: CPLEX vs. Hedging strategies (with local adjustments)

		CPLEX					Tabu-PH			Tabu-HC		
Corr.	$\|\mathcal{S}\|$	L.B.	U.B.	Sol.	Gap	Time	Val.	Gap	Time	Val.	Gap	Time
0	16	410810	458704	31	4.31%	157.47	424523	1.60%	71.31	424656	1.67%	61.00
	32	403121	471938	27	8.46%	194.85	456010	5.46%	139.36	455693	5.44%	136.32
	64	385601	484601	22	15.94%	256.35	464683	10.82%	185.66	464305	10.69%	185.03
0.2	16	414214	459218	30	4.10%	159.94	428347	1.59%	59.45	428439	1.62%	52.09
	32	406689	473296	26	7.89%	196.47	453815	4.80%	141.40	453466	4.74%	126.27
	64	387049	488098	24	16.10%	249.54	462557	10.53%	187.95	460963	10.25%	187.34
0.8	16	408172	458136	31	4.62%	144.81	423587	1.71%	69.62	423743	1.72%	66.35
	32	407050	476565	27	8.31%	195.38	459702	5.39%	139.64	459145	5.37%	129.63
	64	391023	487397	22	14.38%	270.52	469970	10.11%	195.07	470405	10.28%	190.62

Further analysis of the results of Table 3 indicates that the correlation level between scenarios has very little impact overall on the difficulty to address the instances, while increasing the number of scenarios $|\mathcal{S}|$ makes the problems much more difficult for all solution techniques. In 500 minutes of CPU time, CPLEX is able to solve 2/3 of the 120 instances generated for each correlation level, but no more than 60% of those with 64 scenarios. Tabu-PH and Tabu-HC perform significantly better than CPLEX independently of the correlation level and the number of scenarios; their performance is quite similar with a slight advantage for Tabu-HC for the instances with 64 scenarios. One may be concerned about the rather large gaps observed for the problems with 64 scenarios: around 15% for CPLEX and 10% for Tabu-PH and Tabu-HC. One possible explanation for this might be the poor quality of the lower bounds returned by CPLEX, but this issue warrants further investigation. Another issue that warrants further investigation is the spread of running times for any given (Corr., $|\mathcal{S}|$) combination: it is surprising to notice that a third of all instances are not solved by CPLEX in 500 minutes, but that the average running time of CPLEX for all instances is below 220 minutes. This clearly indicates that some instances are solved rather quickly and that there is a huge variability in running times.

Table 5: Aggregated results set **R**: Corr: 0.2, CPLEX vs. Hedging strategies (without local adjustments)

		CPLEX					Tabu-PH			Tabu-HC		
ratio	$\lvert\mathcal{S}\rvert$	L.B.	U.B.	Sol.	Gap	Time	Val.	Gap	Time	Val.	Gap	Time
	16	157719	158019	7	0.06%	63.67	158111	0.09%	5.75	158523	0.22%	6.86
1	32	157019	168161	7	2.07%	66.20	158258	0.23%	47.94	158355	0.28%	27.80
	64	151662	168563	7	3.39%	80.02	167127	3.11%	67.85	158083	1.31%	66.99
	16	371528	522747	5	13.30%	234.88	413942	4.28%	88.16	413942	4.29%	78.99
3	32	354420	544883	4	20.82%	287.27	492793	13.59%	222.55	491877	13.31%	198.99
	64	311073	579702	3	41.51%	334.19	504508	25.47%	269.31	505724	25.80%	269.35
	16	331942	392183	5	5.70%	240.68	346561	2.09%	121.171	346561	2.09%	118.10
5	32	314445	412844	4	12.92%	266.24	386151	7.52%	227.087	386151	7.52%	227.15
	64	286783	442630	4	28.14%	350.06	402844	17.80%	263.033	402844	17.80%	263.06
	16	351726	352379	7	0.19%	68.79	352370	0.27%	30.13	352370	0.27%	18.22
7	32	355225	355929	7	0.21%	90.81	355917	0.20%	70.66	355917	0.20%	70.87
	64	352889	356607	7	1.13%	129.23	354837	0.59%	90.84	354857	0.64%	92.15
	16	858157	870762	6	1.26%	191.68	870750	1.23%	66.19	870750	1.23%	39.87
9	32	852338	884663	4	3.41%	271.85	875958	2.47%	127.53	875972	2.68%	140.12
	64	832838	892986	3	6.33%	354.22	883504	5.71%	248.03	883305	5.69%	250.13

To further investigate this issue, we now report results on the instances with correlation level 0.2 in Table 5. In this Table, results on each line correspond to averages for each of the 8 instances with a given fixed cost and capacity ratio (indicated in the column "ratio") and the same number of scenarios $|\mathcal{S}|$. As expected, this table shows a great variability in the average running times for different groups of problems. It is surprising to observe that the instances that are the most difficult to solve are not the ones with the highest fixed cost and capacity ratio, an observation that is often made for deterministic CMND problems, but those with intermediate ratios. While it is extremely satisfying to note that Tabu-PH and Tabu-HC achieve excellent results in short running times for the easier groups of instances (e.g., those with ratios of 1, 7 or 9, and 32 scenarios or less), it is probably more significant to highlight their substantial superiority over CPLEX on the very difficult instances. In order to do so, we now report in Table 6 the results obtained for the 8 instances with ratio 3 and 64 scenarios. As can be seen from the Table, three of the 8 instances (r04, r07 and r08) are solved to optimality by CPLEX and the hedging strategies rather quickly. It is interesting to note that Tabu-PH and Tabu-HC are able to find the optimal solutions of r04 and r07 in less than one minute, which clearly shows that these are "easy" problems, and in 21 minutes for the significantly more difficult r08, compared to 138 minutes for CPLEX. On the r05 instance, the hedging strategies find a slightly better solution than CPLEX and they take only a quarter of the time that it spends. The more spectacular results are obtained on instances r09 for which both hedging strategies find a solution whose cost is less than 64% of the cost of the CPLEX solution. Tabu-PH and Tabu-HC also clearly outperform CPLEX on instance r10 and r11. Regarding this last instance, one must not be misled by the huge gap recorded for the hedging strategies: this is clearly an instance whose lower bound is very poor.

Globally, the hedging strategies are able to find excellent solutions rather quickly for the easier instances and much better ones than CPLEX for the more difficult ones. In fact, it seems quite obvious

Table 6: Detailed results set **R**: Corr: 0.2, ratio 3 and $|\mathcal{S}| = 64$, CPLEX vs. Hedging strategies (without local adjustments)

	CPLEX				Tabu-PH			Tabu-HC		
Problem	L.B.	U.B.	Gap	Time	Val.	Gap	Time	Val.	Gap	Time
r04.3-0.2-64	57062.9	57062.9	0.00%	8.72	57062.9	0.00%	0.43	57062.9	0.00%	0.44
r05.3-0.2-64	182387	193001	5.82%	500.08	191284	4.88%	125.37	191284	4.88%	125.40
r06.3-0.2-64	430373	502238	16.70%	500.15	487600	13.30%	500.94	487600	13.30%	500.93
r07.3-0.2-64	55513.8	55518.2	0.01%	25.90	55518.2	0.01%	0.37	55518.2	0.01%	0.38
r08.3-0.2-64	150400	150404	0.00%	138.28	150404	0.00%	21.19	150404	0.00%	21.12
r09.3-0.2-64	310667	582334	87.45%	500.10	371875	19.70%	501.07	371875	19.70%	501.09
r10.3-0.2-64	366978	549234	49.66%	500.12	451527	23.04%	501.77	461259	25.69%	502.16
r11.3-0.2-64	935206	2.54782e+06	172.43%	500.20	2.27079e+06	142.81%	503.31	2.27079e+06	142.81%	503.25

that resorting to CPLEX is not an effective solution approach for the larger and more difficult instances. While it is not possible to assess exactly the quality of the solutions produced by Tabu-PH and Tabu-HC on these instances, the huge gap observed between these solutions and the solutions returned by CPLEX suggests that they must be pretty good ones.

7 Conclusions

In this paper, we have developed a series of meta-heuristic solution strategies for stochastic network design problems. These strategies, which are based on progressive hedging principles, were shown to be very efficient to solve a series of stochastic CMND problems when compared to a direct solution approach using the latest version of CPLEX. Given the quality of the results obtained, one can now work towards extending and refining the proposed solution approach. In doing so, an interesting avenue of research would be to further improve the way consensus is driven within the algorithms by using the progressive hedging strategy in a recursive fashion (i.e., where the decomposition approach is applied to subsets of scenarios, which are themselves tackled by a progressive hedging-based solution method). Furthermore, from a general methodological perspective, developing more involved parallel solution strategies also appears promising within the present context, since one could possibly exploit more fully the decomposition of the second-stage problem according to scenarios. We plan to investigate these avenues in the near future.

Acknowledgments

While working on this project, the first author was the NSERC Industrial Research Chair in Logistics Management, ESG UQAM, and Adjunct Professor with the Department of Computer Science and Operations Research, Université de Montréal, and the Department of Economics and Business Administration, Molde University College, Norway. The first four authors were also members of CIRRELT - the Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation.

Partial funding for the project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grants programs, by the contributions of the partners of the Chair, by the Canada Foundation for Innovation, through its Leaders Opportunity Fund, the Ministère de l'éducation, du loisir et du sport du Québec, SUN Microsystems, ILOG, and UQAM, and Grant 171007/V30 from The Research Council of Norway.

References

[1] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming.* Springer, 1997.

[2] I. Ghamlouche, T.G. Crainic, and M. Gendreau. Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51(4):655–667, 2003.

[3] K. K. Haugen, A. Løkketangen, and D. L. Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132:116–122, 2001.

[4] A. Hoff, A.-G. Lium, A. Løkketangen, and T. G. Crainic. A metaheuristic for stochastic service network design. Technical report, Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT-2007-623), C.P. 6128, succursale Centre-ville Montréal QC H3C 3J7 Canada, 2007.

[5] K. Høyland, M. Kaut, and S. W. Wallace. A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24(2-3):169–185, 2003.

[6] A. J. Kleywegt, A. Shapiro, and T. Hommem-De-Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2001.

[7] A. Løkketangen and D. L. Woodruff. Progressive hedging and tabu search applied to mixed integer (0,1) multistage stochastic programming. *Journal of Heuristics*, 2:111–128, 1996.

[8] T. L. Magnanti and R. T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1986.

[9] M. Minoux. Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks*, 19:313–360, 1986.

[10] R. T. Rockafellar and R. J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[11] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167:96–115, 2005.