0-1 Reformulations of the Multicommodity Capacitated Network Design Problem

Antonio Frangioni
Dipartimento di Informatica, Università di Pisa
Largo B. Pontecorvo 1, 56127 Pisa – Italy and
Polo Universitario della Spezia
Via dei Colli 90, 19121 La Spezia – Italy
frangio@di.unipi.it

Bernard Gendron
Département d'informatique et de recherche opérationnelle
Centre de recherche sur les transports
Université de Montréal
C.P. 6128, succ. Centre-ville, Montreal, Quebec H3C 3J7
bernard@crt.umontreal.ca

Abstract

We study 0-1 reformulations of the multicommodity capacitated network design problem, which is usually modeled with general integer variables to represent design decisions on the number of facilities to install on each arc of the network. The reformulations are based on the multiple choice model, a generic approach to represent piecewise linear costs using 0-1 variables. This model is improved by the addition of extended linking inequalities, derived from variable disaggregation techniques. We show that these extended linking inequalities for the 0-1 model are equivalent to the residual capacity inequalities, a class of valid inequalities derived for the model with general integer variables. In this paper, we compare two cutting-plane algorithms to compute the same lower bound on the optimal value of the problem: one based on the generation of residual capacity inequalities within the model with general integer variables, and the other based on the addition of extended linking inequalities to the 0-1 reformulation. To further improve the computational results of the latter approach, we develop a *column-and-row generation* approach; the resulting algorithm is shown to be competitive with the approach relying on residual capacity inequalities.

Keywords: *Multicommodity Capacitated Network Design, Reformulation, Valid Inequalities, Cutting-Plane Method, Column Generation.*

1 Introduction

The *multicommodity capacitated network design problem* (MCND) we consider is defined on a directed network $G = (N, A)$, where N is the set of nodes and A is the set of arcs. We must satisfy the communication demands between several origin-destination pairs, represented by the set of commodities K. For each commodity k, we denote by d^k the positive demand that must flow between the origin $O(k)$ and the destination $D(k)$. While flowing along an arc (i, j), a communication consumes some of the arc capacity; the capacity is obtained by installing on some of the arcs any number of *facilities* of a single type. Installing one facility on arc (i, j) provides a positive capacity u_{ij} at a (nonnegative) cost f_{ij}. A nonnegative routing cost c^k_{ij} also has to be paid for each unit of commodity k moving through arc $(i, j) \in A$. The problem consists of minimizing the sum of all costs, while satisfying demand requirements and capacity constraints. Several applications in transportation, logistics, telecommunications, and production planning can be represented as variants of this classical network design problem [7, 22, 34, 47, 48].

Defining nonnegative *flow variables* x^k_{ij}, which represent the fraction of the flow of commodity k on arc $(i, j) \in A$ (i.e., $d^k x^k_{ij}$ is the flow of commodity k on arc (i, j)) and general integer *design variables* y_{ij}, which define the number of facilities to install on arc (i, j), the problem can then be formulated as the following mixed-integer programming (MIP) model, which we denote I,

$$\min \sum_{k\in K}\sum_{(i,j)\in A} d^k c^k_{ij} x^k_{ij} + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{1}$$

$$\sum_{j\in N^+_i} x^k_{ij} - \sum_{j\in N^-_i} x^k_{ji} = \delta^k_i \qquad i \in N,\ k \in K \tag{2}$$

$$\sum_{k\in K} d^k x^k_{ij} \le u_{ij} y_{ij} \qquad (i, j) \in A \tag{3}$$

$$0 \le x^k_{ij} \le 1 \qquad (i, j) \in A,\ k \in K \tag{4}$$

$$y_{ij} \ge 0 \qquad (i, j) \in A \tag{5}$$

$$y_{ij}\ integer, \qquad (i, j) \in A \tag{6}$$

where $N^+_i = \{j \in N | (i, j) \in A\}$, $N^-_i = \{j \in N | (j, i) \in A\}$, and $\delta^k_i = 1$ if $i = O(k)$, $\delta^k_i = -1$ if $i = D(k)$, and $\delta^k_i = 0$ otherwise.

Although MCND may appear to be a crude approximation of the complex network design problems that arise in practice, it is a flexible modeling tool that can be adapted to a large number of application areas. In particular, various extensions have been studied, most notably in transportation [11, 29, 38, 43] and telecommunications [12, 16, 18]. One of these extensions is the *unsplittable* (or nonbifurcated) variant of the problem, where the flow of each commodity is required to follow one route between the origin and the destination. These constraints require the flow variables to be binary, substantially increasing the difficulty of the problem. Another extension is the *capacity expansion* variant of the problem, where some of the arcs already have an existing capacity. In this case, the network has to be expanded, rather than designed from scratch.

An important special case of MCND is the *network loading problem* (NL), where a limited number (generally, one or two) of facility types, each with given unit cost and capacity, are available on all arcs. Several algorithms have been proposed to solve NL, in particular heuristic approaches to solve large-scale instances (with more than two facility types) [1, 14, 35] and branch- and-cut (B&C) methods based on polyhedral analysis [2, 9, 15, 17, 19, 39, 44, 45, 46, 49]. These B &C algorithms all make use of the so-called *cutset inequalities*, which state that there should be enough capacity installed to satisfy the demands that must be routed across any cutset. Extensions of these inequalities have also been studied, in particular the *metric inequalities*, which specify necessary and sufficient conditions for the feasibility of any multicommodity capacitated network flow problem [21, 32, 33]. The latter are especially interesting, as when there are no routing costs, they can be used to project out the flow variables, thus simplifying the model.

Another interesting variant of MCND occurs when we restrict the design variables to 0-1 values, thus obtaining the *fixed- charge* MCND (FMCND). Research on this problem has focused on heuristic methods [24, 25, 36, 37], bounding procedures and exact approaches [20, 23, 34, 40, 42, 53]. Some conclusions from these studies are useful as well for the MCND. In particular, it is noteworthy that the linear programming (LP) relaxation of FCMND provides an extremely weak approximation of the MIP model. However, by appending the following simple forcing constraints:

$$x_{ij}^k \leq y_{ij}, \quad (i,j) \in A,\, k \in K, \tag{7}$$

called *strong linking inequalities*, we obtain a significantly improved LP lower bound. Also, bounding methods based on the Lagrangian relaxation of constraints (2) have been shown to perform well in practice. It is interesting to note that the convex hull of solutions to the resulting Lagrangian subproblem is obtained by adding the strong linking inequalities (7).

These inequalities are obviously valid for the MCND, but they are weak in this case, as they are not forcing constraints. The *residual capacity inequalities*, introduced by Magnanti, Mirchandani and Vachani [45] and later studied by Atamtürk and Rajan [4] and Atamtürk and Günlük [3], generalize the strong linking inequalities, in the sense that they describe the convex hull of solutions to the subproblem obtained from the Lagrangian relaxation of (2). An alternative to compute the same bound as the Lagrangian dual resulting from this relaxation is to append the residual capacity inequalities to the model, obtaining formulation I^+, and solve the LP relaxation of I^+. One difficulty with this approach is the *exponential* number of residual capacity inequalities. Hence, researchers have used residual capacity inequalities within iterative cutting-plane methods [4, 46]. The success of such approaches depends on the ability to solve the *separation problem*: given a solution to the current LP relaxation, is it possible to find efficiently violated residual capacity inequalities? Atamtürk and Rajan [4] provide a positive answer to this question. We recall this result in Section 2, which reviews the state-of-the-art on the residual capacity inequalities.

An alternative to I, the model with general integer design variables, is to view MCND as a nonlinear nonconvex minimization problem. In particular, the design costs can be modeled using piecewise linear staircase functions, one for each arc, the number of facilities installed on the arc corresponding to a "step", or *segment*, of this function. One can attempt to solve directly the resulting nonlinear model [52], or first reformulate it as a 0-1 MIP model [51], using any of the classical techniques to model piecewise linear functions with 0-1 variables [26,

41]. One of these 0-1 reformulations, the *multiple choice model*, hereafter denoted B, has been studied in the context of multicommodity network flow problems [6, 27, 28]. These studies show that, by defining additional flow variables, one can append to the LP relaxation a class of forcing constraints called the *extended linking inequalities* to obtain high-quality lower bounds, at the expense of solving significantly larger models. Croxton, Gendron and Magnanti [28] also show that these extended linking inequalities describe the convex hull of solutions to the Lagrangian subproblem obtained after relaxing the flow conservation equations.

In this paper, we prove the equivalence between the residual capacity inequalities for model I and the extended linking inequalities for model B, in the sense that the LP relaxations of I^+ and B^+ (B with the extended linking inequalities added) provide the same bound. Since B^+ has a large number of variables and constraints, we develop a *column-and-row generation* procedure to solve its LP relaxation. We show that the resulting approach compares favorably, from a computational point of view, with the cutting-plane method based on the separation of residual capacity inequalities for solving the LP relaxation of I^+ [2]. Thus, a proper 0-1 reformulation of MCND might allow to improve the performances of exact and approximate approaches for the problem.

The structure of the paper is as follows. In Section 2, we review the state-of-the-art on the residual capacity inequalities. In Section 3, we present the 0-1 multiple choice reformulation of the problem and show its relationships with model I, in particular the equivalence between the extended linking inequalities and the residual capacity inequalities. Then, in Section 4, we describe the column-and-row generation procedure that we developed to efficiently optimize over the large-scale 0-1 reformulation. In Section 5, we present the computational results we have performed to compare this method with the cutting-plane approach based on separating residual capacity inequalities [2]. Finally, in Section 6, we draw some conclusions and identify avenues for future research.

Throughout the paper, we use the following notation. For any model M, we denote as $v(M)$, $F(M)$, $conv(F(M))$ and $LP(M)$, its optimal value, its feasible set, the convex hull of $F(M)$, and its LP relaxation, respectively. In our analysis, we often consider the Lagrangian relaxation of the flow conservation equations for different models. For any model M, the resulting Lagrangian subproblem and Lagrangian dual are denoted as $LS(M)$ and $LD(M)$, respectively. Finally, we say that two models are equivalent if their optimal values are the same, for any values of the costs.

2 Residual Capacity Inequalities

The residual capacity inequalities apply to any single arc (i, j) and any subset of the commodities $P \subseteq K$. By introducing the following notation: $d^P = \sum_{k \in P} d^k$, $a_{ij}^P = \frac{d^P}{u_{ij}}$, $q_{ij}^P = \lceil a_{ij}^P \rceil$, $r_{ij}^P = a_{ij}^P - \lfloor a_{ij}^P \rfloor$, the residual capacity inequalities can be written concisely as

$$\sum_{k \in P} \{a_{ij}^k (1 - x_{ij}^k)\} \geq r_{ij}^P (q_{ij}^P - y_{ij}), \quad (i, j) \in A,\ P \subseteq K. \tag{8}$$

Atamtürk and Rajan [4] show that the separation problem for these inequalities can be solved efficiently. If we denote by $(\bar{x}, \bar{y})$ a fractional solution (i.e., $\bar{y}_{ij}$ is fractional for at least one arc (i,j)) to $LP(I)$, we first define, for each arc (i,j), the set $P_{ij} = \{k \in K | \bar{x}^k_{ij} > \bar{y}_{ij} - \lfloor \bar{y}_{ij} \rfloor\}$. Then, given $(i,j) \in A$, if the two following conditions are satisfied:

$$\lfloor \bar{y}_{ij} \rfloor < a_{ij}^{P_{ij}} < \lceil \bar{y}_{ij} \rceil ,$$

$$\sum_{k \in P_{ij}} a^k_{ij}(1 - \bar{x}^k_{ij} - \lceil \bar{y}_{ij} \rceil + \bar{y}_{ij}) + \lfloor \bar{y}_{ij} \rfloor (\lceil \bar{y}_{ij} \rceil - \bar{y}_{ij}) < 0 \ ,$$

the residual capacity inequality corresponding to $P = P_{ij}$ is violated by $(\bar{x}, \bar{y})$. Otherwise, there is no violated residual capacity inequality associated to arc (i,j).

The above developments can be summarized in the following proposition [4, Theorem 1]:

Proposition 1 *The separation problem for the residual capacity inequalities can be solved in $O(|A||K|)$.*

Another important result concerning arc residual inequalities has been proven by Magnanti, Mirchandani and Vachani [45]: inequalities (8) characterize the convex hull of solutions to $LS(I)$, the Lagrangian subproblem obtained after relaxing the flow conservation equations (2). Thus, appending all these inequalities to formulation I, which gives model I^+, and performing the Lagrangian relaxation of (2) yields a subproblem $LS(I^+)$ having the integrality property. These results are summarized as follows:

Proposition 2 $F(LP(LS(I^+))) = conv(F(LS(I)))$.

By classical results from Lagrangian duality theory (see for instance [31]), $LD(I)$, which is thus equivalent to $LD(I^+)$, provides the same bound as the LP relaxation of I^+:

Proposition 3 *$LP(I^+)$ and $LD(I)$ are equivalent.*

We use this result to show the equivalence between the residual capacity inequalities and the extended linking inequalities for model B, presented in the next section.

3 Multiple Choice 0-1 Reformulation

Since we assume that $f_{ij} \geq 0$ for each $(i,j) \in A$, we have

$$y_{ij} \leq T_{ij} = \left\lceil \frac{\sum_{k \in K} d^k}{u_{ij}} \right\rceil .$$

If we denote by $S_{ij} = \{1, \ldots, T_{ij}\}$ the set of possible *nonzero* values of y_{ij}, MCND can be reformulated by introducing $2\sum_{(i,j) \in A} T_{ij}$ *auxiliary variables*, two for each $s \in S_{ij}$ and $(i,j) \in A$, with the following intended meaning:

$$y^s_{ij} = \begin{cases} 1 & \text{if } y_{ij} = s, \\ 0 & \text{otherwise,} \end{cases}$$

$$z_{ij}^s = \begin{cases} \sum_{k \in K} d^k x_{ij}^k, & \text{if } y_{ij} = s \\ 0 & \text{otherwise.} \end{cases}$$

We can then rewrite the problem by appending to (1)–(6) the following additional constraints, obtaining model B:

$$y_{ij} = \sum_{s \in S_{ij}} s y_{ij}^s \qquad (i,j) \in A \tag{9}$$

$$\sum_{k \in K} d^k x_{ij}^k = \sum_{s \in S_{ij}} z_{ij}^s \qquad (i,j) \in A \tag{10}$$

$$(s-1) u_{ij} y_{ij}^s \leq z_{ij}^s \leq s u_{ij} y_{ij}^s \quad (i,j) \in A,\ s \in S_{ij} \tag{11}$$

$$\sum_{s \in S_{ij}} y_{ij}^s \leq 1 \qquad (i,j) \in A \tag{12}$$

$$y_{ij}^s \geq 0 \qquad (i,j) \in A,\ s \in S_{ij} \tag{13}$$

$$y_{ij}^s \ integer \qquad (i,j) \in A,\ s \in S_{ij}. \tag{14}$$

Note that we can now remove constraints (5) and (6), but also constraints (3), which are implied by (9) and (11). Consequently, we can project out variables y_{ij} and obtain a formulation expressed only in terms of the auxiliary binary variables y_{ij}^s, in addition to the flow variables. This formulation corresponds to the so-called *multiple choice* model [26], which can also be derived by interpreting the problem as a multicommodity flow formulation with piecewise linear design costs, each segment of the corresponding cost function on any given arc representing the number of facilities to install on this arc [28]. Using this interpretation, one can also derive two other textbook formulations for piecewise linear cost functions, the so-called incremental and convex combination models [26]. These three formulations are not only equivalent in terms of MIP, but also their LP relaxations provide the same bound. As in [27, 28], we study the multiple choice model, as it lends itself nicely to the addition of simple valid inequalities derived from *variable disaggregation* techniques.

These techniques are based on the addition of the following *extended auxiliary variables*:

$$x_{ij}^{ks} = \begin{cases} x_{ij}^k & \text{if } y_{ij} = s, \\ 0 & \text{otherwise,} \end{cases} \qquad s \in S_{ij},\ k \in K,\ (i,j) \in A.$$

Thus, the original variables in the model can be expressed in terms of these new (and extremely many) variables, through the following definitional equations:

$$x_{ij}^k = \sum_{s \in S_{ij}} x_{ij}^{ks} \qquad (i,j) \in A,\ k \in K, \tag{15}$$

$$z_{ij}^s = \sum_{k \in K} d^k x_{ij}^{ks} \qquad (i,j) \in A,\ s \in S_{ij}. \tag{16}$$

It is easy to see that the LP relaxation of the model obtained from B by adding equations (15)-(16) gives the same bound as that provided by (the more compact) $LP(B)$. However, the following *extended linking inequalities*:

$$x_{ij}^{ks} \leq y_{ij}^s \qquad (i,j) \in A,\ k \in K,\ s \in S_{ij} \tag{17}$$

are redundant in model B but *not* in its LP relaxation, $LP(B)$. We denote as B^+ the model obtained from B by adding (15)-(17). Note that constraints (9)-(17) are redundant for model I, which implies that I and B^+ are equivalent models.

We now prove, by establishing several intermediate results, that the LP relaxation of B^+, $LP(B^+)$, provides the same bound as $LP(I^+)$. The first result follows immediately from the equivalence between models $LS(I)$ and $LS(B^+)$, which itself can be established in a similar way as the equivalence between formulations I and B^+, since constraints (9)-(17) are redundant for $LS(I)$. As both Lagrangian subproblems, $LS(I)$ and $LS(B^+)$, are obtained by relaxing the same constraints, we thus have:

Proposition 4 *$LD(I)$ and $LD(B^+)$ are equivalent.*

We can now compare the LP relaxations of the two formulations with the Lagrangian relaxation of the flow conservation equations, by using the following result, which can be derived from the proof of Theorem 5 in [28]:

Proposition 5 *$F(LP(LS(B^+))) = conv(F(LS(B^+)))$.*

Proof. We first note that $LS(B^+)$ decomposes into $|A|$ subproblems, one for each arc. Hence, we focus on any of these arc-based subproblems and, for convenience, we drop the index (i,j) corresponding to the arc. Let $x = (x^{ks})$ and $y = (y^s)$ denote vectors of extended auxiliary flow variables and auxiliary design variables, respectively. The associated polyhedron P for the arc-based subproblem is therefore defined by the following constraints:

$$\begin{aligned}
(s-1)uy^s \leq \sum_{k\in K} d^k x^{ks} \leq suy^s \quad & s \in S \\
0 \leq x^{ks} \leq y^s \quad & s \in S\ ,\ k \in K \\
\sum_{s\in S} y^s \leq 1 \quad & \\
y^s \geq 0 \quad & s \in S
\end{aligned}$$

To obtain the desired result, we now show that for every extreme point of P, $y^s \in \{0,1\}$ for all $s \in S$. If not, then let $(\hat{x},\hat{y})$ be an extreme point of P with at least one fractional component. Assume that $0 < \hat{y}^r < 1$, for $r \in R \neq \emptyset$. Define the following $|R|+1$ points in P: $(x(0),y(0)) = (0,0)$ and $(x(r),y(r))$, for $r \in R$, with $x^{kr}(r) = \hat{x}^{kr}/\hat{y}^r$, $x^{ks}(r) = 0$, $s \neq r$, $y^r(r) = 1$ and $y^s(r) = 0$, $s \neq r$. Then $(\hat{x},\hat{y}) = (1-\sum_{r\in R}\hat{y}^r)(x(0),y(0)) + \sum_{r\in R}\hat{y}^r(x(r),y(r))$ is a representation of $(\hat{x},\hat{y})$ as a convex combination of $|R|+1 \geq 2$ distinct points in P, contradicting the hypothesis that it is an extreme point of P. ■

From standard Lagrangian duality theory, we then immediately obtain:

Proposition 6 *$LP(B^+)$ and $LD(B^+)$ are equivalent.*

We now only need to recall the result of Proposition 3 to obtain, as an immediate consequence of Propositions 4 and 6:

Theorem 7 *$LP(B^+)$ and $LP(I^+)$ are equivalent.*

Thus, the improvement in the LP relaxation bound provided by formulation B^+ is equivalent to the addition of the residual capacity inequalities to formulation I. This result implies that there are two different ways for achieving the same lower bound:

1. Solve model $LP(I^+)$, with a *polynomial number of variables* ($O(|A||K|)$), but *exponentially many constraints*;
2. Solve model $LP(B^+)$, which has a *pseudo-polynomial number of both variables and constraints*.

While the first alternative can be computed by a standard cutting- plane (or *row generation*) procedure [4], the second option requires to generate *both* rows *and* columns; this contrasts with standard column- *or* row-generation approaches, which customarily assume the other dimension of the problem to be fixed. However, in our case the *same* pricing problem can be used to generate both columns and rows, with the procedure illustrated in the next Section.

4 Column-and-Row Generation Method

In the following, we consider a slightly simplified (but equivalent) form of model B^+, using only the y^s_{ij} and x^{ks}_{ij} variables, obtained by projecting out the original variables x^k_{ij} and y_{ij}, as well as z^s_{ij}, from the formulation using constraints (9), (15) and (16); this boils down to

$$\min \sum_{k\in K}\sum_{(i,j)\in A} d^k c^k_{ij} \sum_{s\in S_{ij}} x^{ks}_{ij} + \sum_{(i,j)\in A} f_{ij} \sum_{s\in S_{ij}} s y^s_{ij} \tag{18}$$

$$\sum_{j\in N^+_i}\sum_{s\in S_{ij}} x^{ks}_{ij} - \sum_{j\in N^-_i}\sum_{s\in S_{ji}} x^{ks}_{ji} = \delta^k_i \qquad i\in N,\, k\in K \tag{19}$$

$$(s-1)u_{ij}y^s_{ij} \le \sum_{k\in K} d^k x^{ks}_{ij} \le s u_{ij} y^s_{ij} \qquad (i,j)\in A,\, s\in S_{ij} \tag{20}$$

$$0 \le x^{ks}_{ij} \le y^s_{ij} \qquad (i,j)\in A,\, k\in K,\, s\in S_{ij} \tag{21}$$

with the additional constraints (12)-(14). Because this formulation has far too many variables and constraints to be solved at once, we use a standard *active set* strategy: the vast majority of the variables being likely to assume value zero at optimality, we select a (small) subset of them, allowed to be positive, and we fix all the other variables to zero. In particular, the forcing constraints (21) ensure that all x^{ks}_{ij} corresponding to some $y^s_{ij} = 0$ also assume value zero. We can thus consider a set $\mathcal{B}$ of *segments*, i.e., pairs $((i,j),s)$ identifying a (small) subset of the y variables. We then define model $B^+_{\mathcal{B}}$, obtained from B^+ by removing all variables y^s_{ij} (and therefore all x^{ks}_{ij}) corresponding to all segments *not* in $\mathcal{B}$.

Clearly, $B^+_{\mathcal{B}}$ is a restriction of B^+. We assume that $LP(B^+_{\mathcal{B}})$ is always feasible: this assumption can be easily satisfied, provided we ensure that in the restricted network, there exists at least one path connecting the origin to the destination for each commodity. Consequently, $v(LP(B^+_{\mathcal{B}}))$ provides a finite upper bound on $v(LP(B^+))$. We are interested in determining whether this upper bound is exact, and if not, in revising $\mathcal{B}$ in such a way that it eventually becomes so.

Achieving these goals is surprisingly simple. Indeed, we need only to consider the Lagrangian subproblem $LS(B^+)$, which can be solved efficiently, as we will see shortly, by the algorithm presented in Section 4.1. Assuming the *restricted master problem* $LP(B^+_{\mathcal{B}})$ has been solved, we denote by (x^*, y^*) its optimal primal solution and by λ^* the optimal values of the dual variables associated to constraints (19). Using λ^* as the values of the Lagrangian multipliers, consider the Lagrangian subproblem $LS(B^+)$, and let $(\bar{x}, \bar{y})$ be an optimal solution to this subproblem. We then have the following result:

Theorem 8 *Either (x^*, y^*) is optimal for $LP(B^+)$, or there must be at least one segment $((i,j),s) \notin \mathcal{B}$ such that $\bar{y}^s_{ij} > 0$.*

Proof. By Proposition 6, the Lagrangian subproblem $LS(B^+)$ provides a lower bound on the optimal value of $LP(B^+)$, i.e.,

$$v(LS(B^+)) \leq v(LP(B^+)) \leq v(LP(B^+_{\mathcal{B}})).$$

Clearly, if $v(LS(B^+)) = v(LP(B^+_{\mathcal{B}}))$, then (x^*, y^*) is optimal for $LP(B^+)$. Otherwise, we have $v(LS(B^+)) < v(LP(B^+_{\mathcal{B}}))$, and we wish to prove there must be at least one segment $((i,j),s) \notin \mathcal{B}$ such that $\bar{y}^s_{ij} > 0$. It is well- known that (x^*, y^*) is also an optimal solution of *its Lagrangian subproblem* $LS(B^+_{\mathcal{B}})$ using the optimal dual variables λ^* as Lagrangian multipliers [31, Theorem 1.1]. The latter is a restriction of the "true" Lagrangian subproblem $LS(B^+)$ (using the same values for the Lagrangian multipliers). Let us assume that there is no segment $((i,j),s) \notin \mathcal{B}$ such that $\bar{y}^s_{ij} > 0$. This implies that $(\bar{x}, \bar{y})$ is feasible for $LS(B^+_{\mathcal{B}})$. Since $(\bar{x}, \bar{y})$ is the optimal solution of a relaxation of $LS(B^+_{\mathcal{B}})$, $(\bar{x}, \bar{y})$ is another optimal solution of $LS(B^+_{\mathcal{B}})$. But, since $v(LS(B^+_{\mathcal{B}})) = v(LP(B^+_{\mathcal{B}}))$, this contradicts $v(LS(B^+)) < v(LP(B^+_{\mathcal{B}}))$. ■

Thus, we can set up an iterative approach that alternates between solving a restricted master problem and a Lagrangian subproblem, similar to the standard Dantzig-Wolfe approach [31]: at each step, $LP(B^+_{\mathcal{B}})$ is solved, the optimal dual variables λ^* associated to the "complicating" constraints (19) are collected, the corresponding Lagrangian relaxation $LS(B^+)$ is formed, and one optimal solution $(\bar{x}, \bar{y})$ is found. Either all segments $((i,j),s)$ such that $\bar{y}^s_{ij} > 0$ are already in $\mathcal{B}$, and therefore the algorithm terminates, providing an optimal solution to $LP(B^+)$, or a new segment has been found that can be added to $\mathcal{B}$. Note that adding a segment $((i,j),s)$ amounts to adding the variable y^s_{ij}, all the variables x^{ks}_{ij} and all the corresponding constraints to the formulation. Thus, at each iteration, both variables *and* constraints are found that can be added to the restricted master problem. Clearly, after a finite number of iterations the algorithm has to terminate with an optimal solution; in the worst case, all segments will be collected, and $B^+_{\mathcal{B}}$ will simply be the whole B^+.

We call the above a *column-and-row generation method*. It can be seen as a special case of the generic "B&C" paradigm, which from the very beginning [50] has advocated the possibility to increasing both dimensions of the underlying formulation. Indeed, generation of both columns and rows has been used with success, under the name "branch-and-cut-and-price", in several applications, often related to network optimization (see for instance [10]). The typical situation is that generation of rows and columns are two independent processes, which require the solution of *two independent subproblems*, one for pricing (generation of

columns) and one for separation (generation of rows). This is also the case of some of the very few other applications where the "column-and-row (or row-and-column) generation" paradigm has been proposed, such as [56]. Our approach is more reminiscent (although independently conceived) of that in [5] for a time-constrained routing problem, since in both cases *one single* subproblem can be used to generate rows and columns. At least qualitatively, however, our application is different in that our model is constructed for the purpose of improving the LP bound of an existing formulation. More similar in spirit (although, again, independently conceived) is the approach in [54, 55] for the Gilmore-Gomory formulation of the cutting stock problem. In that case, a pseudo-polynomial formulation is obtained in a "classical" way by means of the shortest path reformulation of a knapsack problem, while in our case it emerges from an entirely different pattern. In all the cases (such as the present one) where columns and rows can be generated "simultaneously" with only one subproblem, the form of the valid inequalities is so simple that the separation problems for these, once the columns are generated, is straightforward. Developing a formulation where this feature can be exhibited is clearly nontrivial, as the present example shows, nor it is making it effective. In the next two subsections, we examine the implementation issues related to the solution of the Lagrangian subproblem and the restricted master problem, respectively.

4.1 Solving the Lagrangian Subproblem

As pointed out in Section 3, $LS(I)$ and $LS(B^+)$ are equivalent. Therefore, instead of solving $LS(B^+)$, we consider the more compact formulation $LS(I)$, which is obtained from I by the Lagrangian relaxation of the flow conservation equations (2). The corresponding Lagrangian subproblem decomposes into $|A|$ subproblems, one for each arc, having the following form, once we drop the indices corresponding to the arc [4, 45]:

$$\min \quad \sum_{k \in K} \bar{c}^k x^k + \bar{f} y \tag{22}$$

$$\sum_{k \in K} d^k x^k \leq u y \tag{23}$$

$$0 \leq x^k \leq 1 \qquad k \in K \tag{24}$$

$$l \leq y \leq v \tag{25}$$

$$y \ integer \tag{26}$$

In (22), $\bar{c}^k$ and $\bar{f}$ indicate Lagrangian costs, whose specific dependence on the original data and on the Lagrangian multipliers is omitted for the sake of clarity and generality. Indeed, we purposely avoid to even assume $\bar{f} \geq 0$, which is true when relaxing constraints (2) (as in that case $\bar{f} = f$). In this way, we also covet the case where other constraints (e.g. valid inequalities involving the y variables) are relaxed as well, which might produce negative $\bar{f}$. We also assume general lower and upper bounds $l \geq 0$ and $v \leq \left\lceil \sum_{k \in K} d^k / u \right\rceil$ on y; this slight generalization could be useful when the bound is computed within a branch-and-bound (B&B) algorithm where branching is performed on the y variables.

Two observations allow to solve this problem easily. First, for any fixed value of y, the problem reduces to the LP relaxation of a 0-1 knapsack problem, which can be solved in $O(|K|)$ [8] (although in practice we might prefer solving it in $O(|K| \log |K|)$ through

a sorting algorithm). Second, if we relax the integrality constraint on y and allow the continuous variable y to vary, the optimal value of the corresponding subproblem, $Z(y)$, defines a convex function of y when $\bar{f} > 0$.

As a first step in the solution method, we can eliminate the variables x^k such that $\bar{c}^k \geq 0$. Then, if $\bar{f} \leq 0$, it is clearly optimal to set $y = v$ and to solve the corresponding LP knapsack problem to obtain an optimal solution. Otherwise, if $\bar{f} > 0$, we exploit the convexity of $Z(y)$ to derive an optimal integral solution from an optimal solution to the problem obtained by relaxing both the bounding constraints (25) and the integrality constraints (26). The optimal solution to this relaxation is easily obtained by setting $x^k = 1$ if $-\bar{c}^k \geq f d^k/u$, and $x^k = 0$ otherwise. The optimal value for y is then given by

$$B = \frac{1}{u} \sum_{k \in K \,:\, -\bar{c}^k \geq f d^k/u} d^k \quad .$$

By convexity of $Z(y)$, it follows that an integral optimal solution is either $y^1 = \min\{\lceil B \rceil, v\}$ or $y^2 = \max\{\lfloor B \rfloor, l\}$. It then suffices to solve the LP knapsack problem for each of these two values to obtain an optimal solution. Actually, the two LP knapsack problems can be solved simultaneously to further improve the efficiency of the procedure. The above discussion proves the following:

Proposition 9 *The Lagrangian subproblem $LS(I)$ can be solved in $O(|A||K|)$.*

This improves on the complexity, $O(|A||K| \log |K|)$, of the original algorithm suggested in [4, Proposition 1]. We should point out, however, that this is a theoretical bound; our implementation rather solves the LP knapsack problem with a sorting algorithm, and thus attains the complexity $O(|A||K| \log |K|)$, which is preferable in practice (at least for small values of $|K|$) due to the large constants in the linear-time algorithm of [8].

4.2 Solving the Restricted Master Problem

An initial restricted master problem is obtained by simply solving the LP relaxation of model I, generating the variables associated to the segments corresponding to positive flow values on all arcs; this extremely simple initialization step ensures that the restricted master problem is feasible, since the initial set of segments corresponds to a feasible solution.

After solving the Lagrangian subproblem, a new set of candidate segments is identified; we could then add to the restricted master problem *all* the extended linking inequalities corresponding to the newly added segments at once. Clearly, this is not necessary, since for each newly added segment, only a subset of the associated extended linking inequalities need to be generated. Hence, after adding the variables corresponding to the segments identified by the solution of the Lagrangian subproblem, we solve the restricted master problem by a cutting-plane method, verifying violations of the extended linking inequalities for *all* existing segments, including those generated at the previous iterations. This justifies the "column-and-row" name we have given to our approach, since the generation of new elements in the restricted master problem, which could conceivably be performed *simultaneously*, is actually subdivided into two separate phases: first columns are generated, then valid inequalities are separated.

Feasibility of the restricted master problems, after the first one, can be easily ensured by avoiding to removing segments entirely. Since this may lead to the restricted master problem containing a large set of segments (clearly an unwelcome situation), a periodical cleanup of the restricted master problem where segments are eliminated can be a worthy addition to the method. This is actually possible e.g. by employing a classical trick that has been replicated in many contexts (see for instance [30, Section 5.3] for application to Bundle-type algorithms):

Proposition 10 *The column-and-row generation method finitely terminates even if segments are deleted from $\mathcal{B}$, provided that the following simple rule is followed: each time $v(LP(B^+_{\mathcal{B}}))$* strictly decreases*, all segments can be eliminated, except those corresponding to nonzero entries in the current optimal solution of the restricted master problem.*

Proof. Let $\mathcal{B}'$ be the subset of $\mathcal{B}$ where all segments for which $y^s_{ij} = 0$ in the optimal solution to $LP(B^+_{\mathcal{B}})$ are discarded. Clearly, the optimal solution to $LP(B^+_{\mathcal{B}})$ is still feasible, hence optimal, to $LP(B^+_{\mathcal{B}'})$. Thus, feasibility of the restricted master problem is conserved throughout the execution of the algorithm. Furthermore, the sequence of *objective function values* of the restricted master problems is strictly decreasing (at least if restricted to iterations where removal is performed); together with finiteness of the set of possible $\mathcal{B}$, this guarantees finite termination. ■

To the best of our knowledge, this possibility has not been pointed out before in the context of column-and-row generation methods.

Finally, we note that we could have avoided projecting out the original flow variables. Instead, we could have worked with the variables x^k_{ij} and x^s_{ij}, and generate x^{ks}_{ij} by adding the definitional equations (15)–(16), only when the current solution satisfies $x^s_{ij} > 0$ and $x^k_{ij} > 0$. For all variables generated in this way, it is then possible to verify violations of the corresponding extended linking inequalities and add only the violated ones to the restricted master problem. In the next section, we present computational experiments with a "straighforward" implementation of our approach, that does not include the last two features (eliminating segments and working with the original flow variables). Since this implementation is already competitive, when compared to a similar "straightforward" implementation (not including row cleanup schemes, for instance) of the cutting-plane method to solve $LP(I^+)$, we did not implement these refinements, which could potentially further improve the performances of the approach.

5 Computational Results

In this section, we report our computational experiments aimed at comparing the two different approaches we have studied to compute the same lower bound on the optimal value of MCND:

- apply a cutting-plane algorithm to solve $LP(I^+)$, that is, solve by row generation an LP model with exponentially many constraints and "few" variables, the separation problem for the class of constraints being solvable in $O(|A||K|)$;

- apply the column-and-row generation approach to formulation $LP(B^+)$, an LP with a pseudo-polynomial number of both variables and constraints, the Lagrangian subproblem being solvable in $O(|A||K|)$.

All experiments are performed on an AMD Opteron 250 operating at 2.4 Ghz and equipped with 16 Gb RAM (the operating system is Centos 4.2, Linux). The LP formulations are solved with CPLEX (version 10.0). To solve $LP(I^+)$, we have implemented the separator described in Section 2, while to solve $LP(B^+)$, we use the restricted master problem formulations and the Lagrangian subproblem solution procedure presented in Section 4. In both cases, the initial LP relaxation is $LP(I)$, so that the two methods have the same starting point.

In order to obtain feasible solutions to MCND, we use the final formulations obtained after solving $LP(I^+)$ and $LP(B^+)$, hereafter denoted $UB(I^+)$ and $UB(B^+)$: for each of these two MIP models, the heuristic B&B algorithm of CPLEX is performed at the root node, followed by the execution of the *polishing* option of CPLEX, for a limit of one hour, to further improve the quality of the feasible solutions (see the reference manual of CPLEX 10.0 for more details on this option). It is interesting to note that formulation $UB(B^+)$ might not contain the optimal solution to the problem, as some segments corresponding to positive flow values in an optimal solution might have been "forgotten" when solving the LP relaxation. Clearly, this is not the case for model $UB(I^+)$, which necessarily contains an optimal solution to MCND. Hence, if the two formulations were solved to optimality, we would have $v(UB(I^+)) \leq v(UB(B^+))$. However, because we are seeking heuristic solutions, we might have $v(UB(I^+)) > v(UB(B^+))$ (in fact, this happens more often than not).

To compare the two bounding methods, we use the following performance measures: *CPU*, the CPU time required to compute the lower bound; *CUTS*, the number of cuts generated for each method (residual capacity inequalities when solving $LP(I^+)$, extended linking inequalities when solving $LP(B^+)$); *GAP*, the gap (in %) between the upper bound, $v(UB(I^+))$ or $v(UB(B^+))$, and the lower bound $v(LP^+) = v(LP(I^+)) = v(LP(B^+))$: $100 \times (v(UB(M)) - v(LP^+))/v(LP^+)$, with $M = I^+, B^+$. For each instance, we also measure the improvement, *IMP*, between the initial lower bound, $v(LP(I))$, and the final lower bound: $100 \times (v(LP^+) - v(LP(I)))/v(LP(I))$.

In the absence of appropriate real data available, we perform our experiments on 120 problem instances obtained from a network generator similar to the one described in [23] for the FCMND. When provided with target values for $|N|$, $|A|$, and $|K|$, this generator creates arcs by connecting two randomly selected nodes (no parallel arcs are allowed). It selects commodities by choosing uniformly at random one origin and one destination for each commodity. It also generates the variable costs, capacities, and demands as uniformly distributed over user-provided intervals. The capacities can then be scaled by adjusting the capacity ratio, $C = |A|D/\sum_{a \in A} u_a$, to user-provided values (in this formula, $D = \sum_{k \in K} d^k$ is the total demand flowing through the network). When C equals 1, the average arc capacity $\sum_{a \in A} u_a/|A|$ equals the total demand, and the network is lightly capacitated; it becomes more tightly capacitated as C increases.

To simplify the analysis, we divide the problem instances into four classes of instances, with characteristics described in Table 1. For each class, 8 "basic" instances were generated and tested with 4 different values of C, for a total of 32 instances (except for the huge class,

class	$\|N\|$	$\|K\|$
small	20	40
medium	30	100
large	20	200
huge	30	400

Table 1: Description of instances

which, due to very large running times, was restricted to only $6 \times 4 = 24$ instances). All instances in a class share the same number of nodes and commodities, while the number of arcs may vary. Tables 2, 3, 4 and 5 present the computational results obtained with these four classes of instances. Note that for small instances we do not report the CPU times, since they are always less than one second for both approaches.

From the tables, the following trends can be observed:

- Using the "improved" formulations I^+ and B^+ delivers dramatic improvements in the lower bound (column *IMP*), which tend to become more and more significant as the size of the instances grow: the minimum and average improvements are 4% and 19%, 34% and 77%, 43% and 109%, 63% and 97% for small, medium, large and huge instances, respectively. However, the improvement clearly decreases as the instances become more and more capacitated (C grows). This does not mean that more capacitated instances are necessarily more difficult to solve; more often than not, especially in the large and huge sets, the final gap is smaller for tightly capacitated instances ($C = 16$) than for lightly capacitated ones ($C = 1$).

- The row-and-column generation approach to formulation B^+ is competitive with the standard cutting-plane approach to formulation I^+. In particular, for the large and huge instances, it is almost always better (except for four cases), sometimes by a factor of up to 6. It should be remarked that often, although not always, the column-and-row generation approach tends to be slower on tightly capacitated instances (large C) than on lightly capacitated ones (small C), whereas very often the inverse trend is true for the standard cutting-plane approach. Thus, the column-and-row generation approach looks particularly promising when capacities are not extremely tight, although holding well also if they are.

- Somewhat contrary to what the theoretical arguments would suggest, formulation B^+ is a better starting point for constructing MIP-based heuristics for the MCND than I^+: especially on large and huge instances, the final gap attained by B^+ is almost always better than the one attained by I^+ (the only exception being an instance with $C = 16$, and by a very small margin). Since the lower bound is identical in both cases, this only reflects the quality of the upper bound. Heuristics based on model B^+ are capable of finding solutions with a gap better by a factor of up to three compared with those based on I^+, and typically in less CPU time. In absolute value, the gaps are also interesting for such notoriously difficult problems, considering that a general-purpose MIP heuristic is used: only two large and most of the (positively) huge instances

Problem			I^+		B^+	
$\|A\|$	C	*IMP*	*CUTS*	*GAP*	*CUTS*	*GAP*
230	1	16.63%	114	0.00%	189	0.00%
	4	14.21%	114	0.00%	193	0.00%
	8	11.71%	110	0.29%	195	0.29%
	16	7.82%	98	0.40%	185	0.40%
230	1	50.70%	252	0.03%	289	0.03%
	4	39.52%	231	0.03%	272	0.03%
	8	30.01%	216	0.55%	286	0.69%
	16	18.03%	162	0.83%	255	1.16%
230	1	8.67%	91	0.00%	168	0.00%
	4	7.61%	89	0.00%	169	0.00%
	8	6.59%	85	0.13%	176	0.33%
	16	4.74%	85	0.27%	172	0.27%
230	1	29.76%	159	0.00%	220	0.00%
	4	24.94%	140	0.01%	205	0.01%
	8	19.86%	145	0.88%	214	1.34%
	16	13.31%	126	0.69%	236	0.69%
289	1	18.05%	126	0.00%	196	0.00%
	4	14.94%	117	0.00%	197	0.00%
	8	11.21%	116	0.00%	199	0.00%
	16	6.73%	104	0.34%	204	0.34%
289	1	47.77%	242	0.00%	294	0.00%
	4	37.15%	220	0.00%	272	0.00%
	8	26.81%	205	0.10%	264	0.10%
	16	15.04%	145	0.35%	275	0.35%
289	1	9.57%	99	0.00%	175	0.00%
	4	8.20%	92	0.00%	167	0.00%
	8	6.44%	89	0.00%	167	0.00%
	16	4.10%	78	0.55%	162	0.59%
289	1	32.17%	179	0.00%	251	0.00%
	4	26.01%	162	0.00%	231	0.00%
	8	19.34%	153	0.31%	226	0.31%
	16	11.98%	133	0.50%	249	0.50%

Table 2: Results for small instances

Problem			I^+			B^+		
$\|A\|$	C	IMP	CPU	$CUTS$	GAP	CPU	$CUTS$	GAP
517	1	72.93%	25	1189	0.14%	15	1296	0.14%
	4	65.17%	41	1146	0.14%	18	1264	0.14%
	8	56.04%	31	1047	0.14%	22	1211	0.14%
	16	41.79%	38	888	0.49%	28	1084	0.49%
517	1	185.43%	5425	4630	7.32%	4820	4565	6.60%
	4	139.13%	4872	4260	7.78%	4882	4131	6.81%
	8	102.97%	4791	3539	7.16%	5266	3523	7.09%
	16	63.88%	4430	2308	5.71%	5351	2549	5.86%
517	1	53.48%	8	817	0.06%	6	941	0.06%
	4	48.92%	8	758	0.06%	6	885	0.06%
	8	43.16%	8	712	0.06%	5	862	0.06%
	16	33.79%	6	626	0.35%	7	821	0.35%
517	1	152.75%	4340	3533	4.75%	4305	3566	3.73%
	4	122.42%	4238	3368	4.64%	4621	3391	3.55%
	8	94.83%	4433	2985	4.22%	4631	3012	4.61%
	16	63.40%	4170	2079	3.78%	4698	2300	3.59%
669	1	78.90%	44	1708	0.35%	27	1810	0.00%
	4	69.72%	36	1618	0.00%	29	1700	0.00%
	8	59.46%	38	1484	0.00%	26	1566	0.00%
	16	43.44%	30	1161	0.01%	46	1367	0.01%
669	1	124.12%	4271	3275	0.60%	799	3285	0.30%
	4	103.80%	2245	3122	0.30%	1350	3062	0.37%
	8	83.27%	1204	2710	0.29%	755	2739	0.36%
	16	56.99%	939	2071	0.61%	735	2263	0.63%
669	1	58.96%	23	1145	0.43%	10	1251	0.01%
	4	53.51%	14	1091	0.01%	8	1133	0.01%
	8	46.85%	16	986	0.01%	12	1101	0.01%
	16	36.24%	13	842	0.09%	13	994	0.09%
669	1	94.69%	52	2082	0.00%	39	2177	0.00%
	4	82.26%	47	2002	0.00%	67	2060	0.00%
	8	68.84%	44	1779	0.00%	64	1827	0.00%
	16	49.00%	67	1395	0.10%	119	1504	0.10%

Table 3: Results for medium instances

Problem			I^+			B^+		
$\|A\|$	C	*IMP*	*CPU*	*CUTS*	*GAP*	*CPU*	*CUTS*	*GAP*
229	1	171.99%	16756	6497	18.90%	4970	5456	6.59%
	4	119.97%	18286	6153	11.71%	5254	4889	5.43%
	8	81.44%	13577	5176	7.97%	6110	3886	5.04%
	16	46.35%	6428	3174	5.45%	5110	2730	4.65%
229	1	217.98%	69712	9552	25.31%	5939	7312	10.37%
	4	130.29%	61596	8836	22.42%	6495	6026	12.30%
	8	81.88%	36318	6780	16.77%	6214	4427	9.68%
	16	43.25%	6976	3165	5.23%	6734	2724	6.03%
229	1	157.04%	11953	5633	14.06%	4788	4815	4.63%
	4	115.20%	13682	5394	5.53%	4998	4436	5.11%
	8	80.98%	10423	4599	5.67%	5802	3758	5.03%
	16	47.33%	6001	3088	4.03%	5711	2705	4.33%
229	1	196.29%	38100	8047	25.83%	5738	6476	8.84%
	4	127.29%	39952	7550	18.31%	6187	5481	6.74%
	8	83.23%	23483	5950	15.20%	6043	4207	6.84%
	16	45.99%	7631	3382	6.99%	5847	2749	4.92%
287	1	153.58%	13860	5874	13.53%	5312	5178	5.59%
	4	116.90%	15095	5753	10.12%	5859	4871	5.62%
	8	86.27%	10707	5057	7.11%	5952	4204	6.29%
	16	54.50%	6535	3475	5.81%	5792	3021	4.20%
287	1	199.78%	38899	8720	23.36%	6425	7180	9.31%
	4	133.87%	33748	8315	18.30%	6844	6231	8.81%
	8	91.38%	29183	6639	17.54%	7828	4993	7.92%
	16	50.67%	8172	3686	8.46%	5747	3045	6.70%
287	1	145.78%	12912	5439	9.65%	5423	4908	5.62%
	4	113.30%	11471	5377	9.38%	5396	4630	5.23%
	8	84.88%	9243	4844	9.21%	5723	4118	5.18%
	16	54.51%	6816	3212	3.87%	5808	2944	3.65%
287	1	191.80%	36166	8400	22.27%	6751	6859	8.22%
	4	131.80%	35676	7913	16.65%	7015	6114	8.79%
	8	91.50%	22752	6391	11.44%	7449	4845	7.30%
	16	51.96%	8988	3660	7.40%	5402	3029	6.28%

Table 4: Results for large instances

Problem			I^+			B^+		
$\|A\|$	C	*IMP*	*CPU*	*CUTS*	*GAP*	*CPU*	*CUTS*	*GAP*
519	1	104.71%	42183	9104	16.29%	24293	8174	7.96%
	4	95.94%	39145	8882	14.26%	24489	8023	8.24%
	8	84.99%	39880	8660	12.58%	28748	7773	7.67%
	16	67.33%	37077	8361	11.96%	30645	7153	8.49%
519	1	147.39%	383639	15741	37.61%	56997	12254	17.68%
	4	127.39%	396879	15795	23.23%	63088	11858	22.72%
	8	105.60%	323178	15227	23.46%	68849	11082	22.68%
	16	76.93%	196321	12860	21.57%	66350	9315	23.19%
519	1	91.63%	17864	7313	24.03%	19351	6993	5.72%
	4	85.17%	19828	7323	8.14%	16960	6848	5.42%
	8	76.91%	18344	7159	7.26%	20222	6702	5.70%
	16	62.76%	18717	6960	8.42%	23892	6242	5.29%
519	1	131.28%	192903	12851	56.92%	42195	10698	15.25%
	4	116.39%	148750	12696	22.13%	45282	10319	15.25%
	8	99.03%	138651	12238	21.27%	50533	9749	15.22%
	16	74.44%	114130	10941	18.85%	46412	8503	13.59%
668	1	122.93%	171554	12715	20.13%	76074	10900	15.88%
	4	112.29%	137467	12567	19.58%	66537	10611	18.31%
	8	99.21%	136076	12309	16.90%	81745	10261	19.09%
	16	78.74%	107195	11524	16.46%	96239	9381	15.56%
668	1	108.69%	69891	10487	14.74%	58145	9310	10.85%
	4	100.84%	55764	10444	13.12%	53774	9244	10.64%
	8	90.93%	63318	10114	13.72%	52405	8902	10.17%
	16	74.38%	55629	9687	12.72%	63779	8291	10.42%

Table 5: Results for huge instances

terminate with a gap larger than 10%. It should be noted, however, that the advantage of using formulation B^+ over model I^+ reduces as C increases.

The above observations allow us to conclude that the improved formulations are a worthy starting point for constructing exact and approximate approaches to MCND. The proposed column-and-row generation approach for solving formulation B^+ is especially interesting for not too-tightly capacitated instances (which, however, are the ones where the improved formulations provide the largest impact on the lower bounds) and as the size of the instances grow.

6 Conclusions

We have shown that a nontrivial 0-1 reformulation of MCND provides the same LP bound obtained by adding exponentially many residual capacity inequalities to the LP relaxation of the general integer formulation. This gives two different ways for obtaining the same bound. Our computational experiments show that the "more balanced" master problem of the column-and-row generation approach results in a more efficient approach, especially for large-scale, not too-tightly capacitated instances with many commodities. Interestingly, the 0-1 reformulation of the problem also appears to be a better starting point for constructing MIP-based heuristic approaches to MCND than the standard integer formulation, even if the latter is augmented with residual capacity inequalities.

The obtained results open a number of interesting research lines that we intend to pursue in the future. First and foremost, similar approaches have been applied independently to rather different applications [5, 54, 55, 56], but to date there does not seem to be a general abstract framework which nicely covers them all. We suspect that developing such a framework could prove rewarding, especially if issues like stabilization of the column(-and- row) generation algorithm [13, 30] are taken into account. As far as MCND (or other similar network design problems) is concerned, the presented approach is clearly only a first step towards the development of a complete solution algorithm. While the column-and-row generation approach looks competitive "in isolation," it remains to be seen how the various elements of a B&C approach, such as heuristics, other valid inequalities, and branching schemes, impact on the performances of the different methods for obtaining the same "basic" lower bound. It is only by implementing and refining a complete solution algorithm that these issues will be properly addressed.

Acknowledgments

We thank two anonymous referees, whose comments have helped us write a better paper. We are grateful to Serge Bisaillon for his help with implementing and testing the algorithms. We also gratefully acknowledge financial support for this project provided by NSERC (Canada) and by MIUR (Italy) under the PRIN project "Optimization, simulation and complexity in telecommunication network design and management".

References

[1] Y.K. Agarwal. Design of Capacitated Multicommodity Networks with Multiple Facilities. *Operations Research*, 50:333–344, 2002.

[2] A. Atamtürk. On Capacitated Network Design Cut-Set Polyhedra. *Mathematical Programming*, 92:425–437, 2002.

[3] A. Atamtürk and O. Günlük. Network Design Arc-Set with Variable Upper Bounds. *Networks*, 50:17–28, 2007.

[4] A. Atamtürk and D. Rajan. On Splittable and Unsplittable Flow Capacitated Network Design Arc-Set Polyhedra. *Mathematical Programming*, 92:315–333, 2002.

[5] P. Avella, B. D'Auria, and S. Salerno. A LP-based heuristic for a time-constrained routing problem. *European Journal of Operational Research*, 173:120–124, 2006.

[6] A. Balakrishnan and S. Graves. A Composite Algorithm for a Concave-Cost Network Flow Problem. *Networks*, 19:175–202, 1989.

[7] A. Balakrishnan, T.L. Magnanti, and P. Mirchandani. Network Design. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley and Sons, 1997.

[8] E. Balas and E. Zemel. An Algorithm for Large Zero-One Knapsack Problems. *Operations Research*, 28:1130–1154, 1980.

[9] F. Barahona. Network Design Using Cut Inequalities. *SIAM Journal on Optimization*, 6:823–837, 1996.

[10] C. Barnhart, C.A. Hane, and P. Vance. Using Branch-and-Price-and-Cut to Solve Origin-Destination Integer Multicommodity Flow Problems. *Operations Research*, 48:318–326, 2000.

[11] C. Barnhart, H. Jin, and P.H. Vance. Railroad Blocking: A Network Design Application. *Operations Research*, 48:603–614, 2000.

[12] P. Belotti, L. Brunetta, and F. Malucelli. Multicommodity Network Design with Discrete Node Costs. *Networks*, 49:90–99, 2007.

[13] H. Ben Hamor and J. Desrosiers. Stabilized Column Generation for Highly Degenerate Multiple-Depot Vehicle Scheduling Problems. *Computers & Operations Research*, 33:910–927, 2006.

[14] D. Berger, B. Gendron, J.-Y. Potvin, S. Raghavan, and P. Soriano. Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6:253–267, 2000.

[15] D. Bienstock, S. Chopra, O. Günlük, and C. Tsai. Minimum Cost Capacity Installation for Multicommodity Network Flows. *Mathematical Programming*, 81:177–199, 1998.

[16] D. Bienstock and O. Günlük. Computational Experience with a Difficult Mixed-Integer Multicommodity Flow Problem. *Mathematical Programming*, 68:213–237, 1995.

[17] D. Bienstock and O. Günlük. Capacitated Network Design–Polyhedral Structure and Computation. *INFORMS Journal of Computing*, 8:243–259, 1996.

[18] A. Chabrier, E. Danna, C. Le Pape, and L. Perron. Solving a Network Design Problem. *Annals of Operations Research*, 130:217–239, 2004.

[19] S. Chopra, I. Gilboa, and S.T. Sastry. Source Sink Flows with Capacity Installation in Batches. *Discrete Applied Mathematics*, 85:165–192, 1998.

[20] M. Chouman, T.G. Crainic, and B. Gendron. A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Network Design. Publication crt-316, Centre de recherche sur les transports, Université de Montréal, 2003.

[21] A.M. Costa, J.-F. Cordeau, and B. Gendron. Benders, Metric and Cutset Inequalities for Multicommodity Capacitated Network Design. *Computational Optimization and Applications*, forthcoming, 2007.

[22] T.G. Crainic. Service Network Design in Freight Transportation. *European Journal of Operational Research*, 122:272–288, 2000.

[23] T.G. Crainic, A. Frangioni, and B. Gendron. Bundle-Based Relaxation Methods for Multicommodity Capacitated Fixed Charge Network Design Problems. *Discrete Applied Mathematics*, 112:73–99, 2001.

[24] T.G. Crainic, M. Gendreau, and J. Farvolden. Simplex-Based Tabu Search for the Multicommodity Capacitated Fixed Charge Network Design Problem. *INFORMS Journal on Computing*, 12:223–236, 2000.

[25] T.G. Crainic, B. Gendron, and G. Hernu. A Slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design. *Journal of Heuristics*, 10:525–545, 2004.

[26] K.L. Croxton, B. Gendron, and T.L. Magnanti. A Comparison of Mixed-Integer Programming Models for Non-Convex Piecewise Linear Cost Minimization Problems. *Management Science*, 49:1268–1273, 2003.

[27] K.L. Croxton, B. Gendron, and T.L. Magnanti. Models and Methods for Merge-in-Transit Operations. *Transportation Science*, 37:1–22, 2003.

[28] K.L. Croxton, B. Gendron, and T.L. Magnanti. Variable Disaggregation in Network Flow Problems with Piecewise Linear Costs. *Operations Research*, 55:146–157, 2007.

[29] J.M. Farwolden and W.B. Powell. Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28:256–272, 1994.

[30] A. Frangioni. Generalized Bundle Methods. *SIAM Journal on Optimization*, 13:117–156, 2002.

[31] A. Frangioni. About Lagrangian Methods in Integer Optimization. *Annals of Operations Research*, 139:163–193, 2005.

[32] V. Gabrel, K. Knippel, and M. Minoux. Exact Solution of Multicommodity Network Optimization Problems with General Step Cost Functions. *Operations Research Letters*, 25:15–23, 1999.

[33] V. Gabrel, K. Knippel, and M. Minoux. A Comparison of Heuristics for the Discrete Cost Multicommodity Network Optimization Problem. *Journal of Heuristics*, 9:429–445, 2003.

[34] B. Gendron, T.G. Crainic, and A. Frangioni. Multicommodity Capacitated Network Design. In Soriano, P. and Sansò, B., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academics Publisher, 1999.

[35] B. Gendron, J.-Y. Potvin, and P. Soriano. Diversification Strategies in Local Search for a Nonbifurcated Network Loading Problem. *European Journal of Operational Research*, 142:231–241, 2002.

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

[37] I. Ghamlouche, T.G. Crainic, and M. Gendreau. Path Relinking, Cycle-Based Neighbourhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research*, 131:109–133, 2004.

[38] T. Grünert and H.-J. Sebastian. Planning Models for Long-Haul Operations of Postal and Express Shipment Companies. *European Journal of Operational Research*, 122:289–309, 2000.

[39] O. Günlük. A Branch-and-Cut Algorithm for Capacitated Network Design Problems. *Mathematical Programming*, 86:17–39, 1999.

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

[41] A.B. Keha, I.R. de Farias Jr., and G.L. Nemhauser. Models for Representing Piecewise Linear Cost Functions. *Operations Research Letters*, 32:44–48, 2004.

[42] G. Kliewer and L. Timajev. Relax-and-Cut for Capacitated Network Design. In *Proceedings of Algorithms-ESA 2005: 13th Annual European Symposium on Algorithms*, pages 47–58. Lecture Notes in Computer Science 3369, 2005.

[43] J.M. Leung, T.L. Magnanti, and V. Singal. Routing in Point-to-Point Delivery Systems: Formulations and Solution Heuristics. *Transportation Science*, 24:245–260, 1990.

[44] T.L. Magnanti and P. Mirchandani. Shortest Paths, Single Origin-Destination Network Design, and Associated Polyhedra. *Networks*, 23:103–121, 1993.

[45] T.L. Magnanti, P. Mirchandani, and R. Vachani. The Convex Hull of Two Core Capacitated Network Design Problems. *Mathematical Programming*, 60:233–250, 1993.

[46] T.L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and Solving the Two-Facility Capacitated Network Loading Problem. *Operations Research*, 43:142–157, 1995.

[47] T.L. Magnanti and R.T. Wong. Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, 18:1–55, 1984.

[48] M. Minoux. Network Synthesis and Optimum Network Design Problems: Models, Solution Methods and Applications. *Networks*, 19:313–360, 1989.

[49] P. Mirchandani. Projections of the Capacitated Network Loading Problem. *European Journal of Operational Research*, 122:534–560, 2000.

[50] M.W. Padberg and G. Rinaldi. A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Travelling Salesman Problems. *SIAM Review*, 33:60–100, 1991.

[51] S. Raghavan and S. Stanojević. A Note on Search by Objective Relaxation. In S. Raghavan and G. Anandalingam, editors, *Telecommunications Planning: Innovations in Pricing, Network Design and Management*, chapter 10, pages 181–201. Springer, 2006.

[52] F.S. Salman, R. Ravi, and J. Hooker. Solving the Capacitated Local Access Network Design Problem. Working paper, Purdue University, 2004.

[53] M. Sellmann, G. Kliewer, and A. Koberstein. Lagrangian Cardinality Cuts and Variable Fixing for Capacitated Network Design. In *Proceedings of Algorithms-ESA 2002: 10th Annual European Symposium on Algorithms*, pages 845–858. Lecture Notes in Computer Science 2461, 2002.

[54] J.M. Valério de Carvalho. Exact Solution of Bin-Packing Problems Using Column Generation and Branch-and-Bound. *Annals of Operations Research*, 86:629–659, 1999.

[55] J.M. Valério de Carvalho. LP Models for Bin Packing and Cutting Stock Problems. *European Journal of Operational Research*, 141:253–273, 2002.

[56] E.J. Zak. Row and column generation technique for a multistage cutting stock problem. *Computers & Operations Research*, 29:1143–1156, 2002.