CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

MIP-Based Tabu Search for Service Network Design with Design-Balanced Requirements

Mervat Chouman
Teodor Gabriel Crainic

November 2011

CIRRELT-2011-68

Bureaux de Montréal :

Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :

Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

MIP-Based Tabu Search for Service Network Design with Design-Balanced Requirements

Mervat Chouman[1], Teodor Gabriel Crainic[2,*]

[1] Department of Operation and Information Management, Effat University, Jeddah, Kingdom of Saudi Arabia

[2] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

Abstract. The paper introduces a MIP-TS matheuristic for the design-balanced capacitated multicommodity network design, one of the premier formulations for the service network design problem with asset management concerns increasingly faced by carriers within their tactical planning processes. The matheuristic combines a cutting-plane procedure efficiently computing tight lower bounds and a Tabu Search meta-heuristic exploiting a new cycle-based neighbourhood satisfying the design-balanced requirements. Learning mechanisms embedded into each of these procedures help in fixing variables and identifying good starting and intensification solutions. Extensive computational experiments show the efficiency of the proposed procedures in obtaining high-quality solutions, outperforming the current best methods from the literature.

Keywords. Service network design, design-balanced constraints, tabu search, cutting-plane method, matheuristic.

Acknowledgements. The authors thank Ms. Geneviève Hernu, analyst with the Chair, for her much welcome assistance in conducting the experimental phase of this project, and the CIRRELT for access to the material and human resources of its computing network. Partial funding for this 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 partners of the Chair, CN, Rona, Alimentation Couche-Tard and the Ministry of Transportation of Québec. The computing equipment principally used was acquired with the support of the Canadian Foundation for Innovation and its partners, mainly the Quebec Ministry of Education and UQAM.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

* Corresponding author: TeodorGabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2011

© Copyright Chouman, Crainic and CIRRELT, 2011

1 Introduction

Network design formulations are used to model a wide variety of problems in several fields such as transportation, logistics, distribution, production, etc. Surveys on network design may be found in Magnanti and Wong (1984), Minoux (1989), and Crainic (2000). We are particularly interested in fixed-cost, multicommodity, capacitated formulations characterized by a network with link capacities and a set of known demands between origin-destination nodes. The network design problem then aims to construct a network, by choosing the arcs to be used, and to satisfy the demand, by determining the flow distribution on each arc, at minimum cost. In such problem, in addition to the usual per-unit routing cost, a fixed cost is payed as soon as a link is used.

Service network design belongs to this broad problem class, where links represent "services" to operate within a given system. Service network design is particularly used to address tactical planning issues for consolidation-based transportation carriers (Crainic, 2003; Crainic and Kim, 2007). More precisely, it relates to the decision problem of selecting transportation services to operate over a mid-term planning horizon, together with their frequencies or schedules as well as the main strategies of moving loads through the resulting service network, to optimize the economic and service criteria of the carrier and achieve an efficient allocation and utilization of its resources, given forecast origin-to-destination demand. The result of the tactical planning process usually is a transportation plan and schedule for a given time length, e.g., a day or a week, to be repeatedly operated over the planning horizon of the "next season" (i.e., from a few months to a year). One calls such a schedule periodic and circular.

The management of assets, e.g., power units, vehicles, crews, etc., was generally not detailed in most of the contributions in the literature (Crainic, 2003; Crainic and Kim, 2007), with a few exceptions where the cost of owning and operating particular assets, planes or ships, for example, was dominating the other cost considerations (e.g. Armacost et al., 2002; Smilowitz et al., 2003; Lai and Lo, 2004). Constraints requiring that the same number of assets enters and exists each terminal, called *design-balanced constraints* by Pedersen et al. (2009), and ad-hoc solution methods were proposed. Resource-management considerations in tactical planning processes and models are becoming wide spread, as so-called full-asset-utilization policies aiming to use assets continuously following circular routes Crainic and Kim (2007); Bektas and Crainic (2008) are being adopted by carriers of all modes. Andersen et al. (2009b,a) give an up-to-date review of previous contributions to the field and study formulations for various asset-management considerations within service network design.

In this paper, we focus on the *design-balanced capacitated multicommodity network design* problem (*DBCMND*), a generic network design problem with design-balanced requirements formally introduced in Pedersen et al. (2009), together with a Tabu Search (TS) meta-heuristic. The DBCMND is NP-Hard, as it is a special case of the well-known

NP-Hard multicommodity fixed charge problem (Magnanti and Wong, 1984) and, thus, exact methods reach their limits rather rapidly (Andersen et al., 2011). Moreover, as shown by Pedersen et al. (2009), even feasible solutions are difficult to obtain for the DBCMND.

Our goal is to address these challenges and propose a methodology to identify good-quality feasible solutions for realistically-dimensioned instances. We propose a matheuristic combining an exact lower-bound computing method and a Tabu Search meta-heuristic (Glover, 1986; Glover and Laguna, 1997) searching for good upper-bound solutions. The former method uses the cutting-plane procedure proposed by Chouman et al. (2009, 2011) for the *capacitated multicommodity fixed charge network design* problem (*CMND*) to compute tight lower bounds in a quick computational time. The proposed Tabu Search meta-heuristic is based on the exploration of the space of the arc-design variables satisfying the design-balanced requirements. The neighbourhood structure is identified using a new arc-balanced cycle move. To evaluate the performance of the proposed method, a computational study is performed on a large set of test instances used in the literature.

The contributions of this paper are threefold. First, the introduction of a new design-balanced neighbourhood and the development of an efficient procedure to explore it. The procedure, based on tagging and a labeling shortest path algorithm, closes and opens arcs around cycles in such a way that the resulting neighbours satisfy the arc-balanced constraints. Second, the paper introduces a MIP-based learning process to identify good-quality feasible solutions, which may then be improved by the TS method (or any other method, for this matter). More precisely, it shows how to take advantage of the time spent computing bounds to compile statistics characterizing attributes of already-encountered solutions, which are then used to find quickly feasible solutions. Finally, we introduce a MIP-TS matheuristic integrating these ingredients shown, through computational experiments on a large set of instances with various characteristics, to outperform existing solution methods in solution quality and computational effort.

This paper is organized as follows. We recall the problem formulation in the next section. Section 3 describes the proposed matheuristic. Computational results are reported in Section 4. Conclusions and perspectives are given in Section 5.

2 Problem Formulation

Given a directed graph $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs, and a set of commodities (or origin-destination pairs) $\mathcal{K}$ to be routed according to a known demand w^k for each commodity k, the problem is to satisfy the demand at minimum cost. The cost consists of the sum of transportation costs and fixed design costs, the latter being charged whenever an arc is included in the optimal design. The transportation cost per unit of commodity k on arc (i,j) is denoted $c_{ij}^k \geq 0$, while the fixed design cost for arc (i,j) is denoted $f_{ij} \geq 0$. A limited capacity, denoted u_{ij}, is associated to each arc (i,j). An origin $O(k)$ and a destination $D(k)$ are associated to each commodity k. We introduce continuous flow variables x_{ij}^k, which reflect the amount of flow on each arc (i,j) for each commodity k, and 0-1 design variables y_{ij}, which indicate if arc (i,j) is used or not. With this notation, the mathematical formulation of the Design-Balanced Multicommodity Capacitated Fixed Charge Network Design problem becomes

$$\min_{x,y} \quad \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij}^k, \tag{2.1}$$

$$\sum_{j\in\mathcal{N}_i^+} x_{ij}^k - \sum_{j\in\mathcal{N}_i^-} x_{ji}^k = d^k, \qquad \forall\, i\in\mathcal{N}, \forall\, k\in\mathcal{K}, \tag{2.2}$$

$$\sum_{j\in\mathcal{N}_i^+} y_{ij} - \sum_{j\in\mathcal{N}_i^-} y_{ji} = 0, \qquad \forall i\in\mathcal{N}, \tag{2.3}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij}y_{ij}, \quad \forall\,(i,j)\in\mathcal{A}, \tag{2.4}$$

$$x_{ij}^k \geq 0, \qquad \forall\,(i,j)\in\mathcal{A},\ \forall\, k\in\mathcal{K}, \tag{2.5}$$

$$y_{ij} \in \{0,1\}, \quad \forall\,(i,j)\in\mathcal{A}, \tag{2.6}$$

where $\mathcal{N}_i^- = \{j\in\mathcal{N} : (j,i)\in\mathcal{A}\}$, $\mathcal{N}_i^+ = \{j\in\mathcal{N} : (i,j)\in\mathcal{A}\}$, and

$$d^k = \begin{cases} w^k, & \text{if } i = O(k), \\ -w^k, & \text{if } i = D(k), \\ 0, & \text{otherwise,} \end{cases}$$

The objective function (2.1) minimizes the total cost computed as the sum of the total fixed cost for the arcs included in the optimal design (denoted as *open*) plus the total commodity transportation cost. Constraints (2.2) correspond to the flow conservation equations for each node and each commodity, while Constraints (2.3) are the design-balanced constraints ensuring that the total number of open arcs entering a node is equal to the total number of open arcs leaving that node. Relations (2.4) represent capacity constraints for each arc that also link flow and design variables by forbidding any flow to pass through an arc not already chosen as part of the design.

Note that, the linear relaxation (LP) of this formulation is obtained by replacing the integrality constraints (2.6) by $0 \leq y_{ij} \leq 1,\ \ \forall (i,j)\in\mathcal{A}$. Note also that, removing constraints set (2.3) yields the well known CMND, which is NP-Hard (Magnanti and Wong, 1984; Balakrishnan et al., 1997) and thus makes the DBCMND NP-hard as well. Practically, considerable algorithmic challenges are imposed when solving realistically-sized network problem instances. These challenges are due to the large size of real applications, to the trade-offs to be found between variable and fixed costs, and to the competition among commodities due to the limited capacity on the arcs. In addition, the design-balanced constraints (2.3) link the design choices and thus increase the combinatorial nature of the problem and add to the algorithmic challenges.

3 Methodological Approach

We now present the MIP-TS matheuristic we propose for the DBCMND, motivated essentially by three observations. First, the cutting-plane procedure proposed by Chouman et al. (2009, 2011) has proved effective in computing tight lower bounds for the CMND in quick computational time when compared to state-of-the-art software. It is thus natural to explore the efficiency of the procedure in the context of the DBCMND. Second, various memories characterizing attributes of LP solutions can be built during the cutting-plane procedure and may then be used not only to guide the search but also to identify good-quality feasible solutions. It is noteworthy that this is the first generic procedure providing the means to obtain efficiently good initial solutions for the DBCMND. Third, Tabu Search displayed good performance on several hard problem classes, including network design (Ghamlouche et al., 2003; Pedersen et al., 2009), and it appeared interesting to investigate the effectiveness of TS in identifying good solutions when the search is guided by a MIP-based learning process.

Algorithm 1 illustrates the main components of the proposed matheuristic combining the three algorithmic components. The methods starts, Phase I, by computing lower bounds on the optimal value of DBCMND using the cutting-plane procedure of Chouman et al. (2009, 2011) and compiling a number of statistics on solution characteristics. These memories are used in Phase II to temporarily fix a number of design variables and identify a first feasible solution by solving the restricted problem using a MIP algorithm (e.g., the branch-and-cut of a commercial software). Starting from this solution, Phase III executes the TS meta-heuristic based on a new neighbourhood that preserves the design-balanced requirements of the problem. Memories characterizing good attributes of explored solutions are also built while performing the TS procedure, which continues until no improvement is observed for a given number of iterations. Provided the algorithms is not stopped, Phase IV launches an intensification mechanism by, first, using the memories built by the TS meta-heuristic to temporarily fix a number of design variables and, then, identifying a new feasible solution using a MIP method and restarting the TS procedure. The overall algorithm stops when either the total computational time exceeds a given time-limit or the number of consecutive times the TS is launched from a new solution without improving the solution quality reaches a pre-specified number. The overall MIP-TS matheuristic is detailed in the following four subsections, one for each of its four phases.

Algorithm 1 MIP-TS Matheuristic

Phase I: Lower-bound computation (LBC)
Run the cutting-plane algorithm (Algorithm 2) and compile F, L, and R LBC memories

Phase II: First feasible solution
Perform the α_0-fixing heuristic (Algorithm 3) based on LBC memories
Solve the resulting reduced DBCMND with a MIP method to obtain a starting feasible solution
Let *Best-Solution* and *Current-Solution* be this solution
Stop and return *Best-Solution* if running time exceeds *TimeLimit*

Phase III: Enhance the solution
Improve *Best-Solution* by performing the cycle-based TS meta-heuristic (Algorithm 4) starting from the *Current-Solution*
Compile TS memories while executing the meta-heuristic
if *NbProcessWithoutImprov* $\geq$ *NbIterProcess* or Running time $\geq$ *TimeLimit* **then**
  Stop and
  return *Best-Solution*
end if

Phase IV: Intensification and a new feasible solution
Perform α_1-fixing heuristic (Algorithm 7) based on TS memories
Solve the resulting reduced DBCMND with a MIP method to obtain a new feasible solution
Let *Current-Solution* be this new feasible solution
Go to Phase III

3.1 Lower bound computation

Any valid inequality (VI) that is valid for a relaxation of a problem, is valid for the problem itself. Therefore, as by dropping constraints (2.3) one obtains a CMND, any valid inequality for the CMND is valid for the DBCMND. More precisely, the families of inequalities studied in Chouman et al. (2009) and Chouman et al. (2011) are valid for the DBCMND and can be used to improve the formulation of the problem and strengthen the quality of its LP bounds.

The same studies have shown, however, that different VI families display quite different behaviours relative to their capability to improve the quality of bounds in reasonable computation times. Based on those studies and aiming for a combination of VI yielding a good trade-off between solution quality and time, and, thus, an efficient cutting-plane method, we consider three families of VIs only: the strong, cover, and flow-pack inequalities. We briefly describe the families. More detailed discussions relative to the associated separation problems and implementation issues are to be found in the two references above.

Strong Inequalities (SI) are defined as

$$x_{ij}^k \;\leq\; d^k y_{ij}, \;\; \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}. \tag{3.1}$$

Adding SI to the model significantly improves the quality of the design LP lower bounds (Crainic et al., 1999; Gendron and Crainic, 1994).

Cover Inequalities (CI) are defined in terms of cutsets of the network. Let $\mathcal{S} \subset \mathcal{N}$ be any non-empty subset of $\mathcal{N}$ and $\bar{\mathcal{S}} = \mathcal{N} \backslash \mathcal{S}$ its complement. We identify the corresponding cutset by $(\mathcal{S}, \bar{\mathcal{S}})$, i.e., the set of arcs that connect a node in $\mathcal{S}$ to a node in $\bar{\mathcal{S}}$. Let $d_{(\mathcal{S},\bar{\mathcal{S}})} = \sum_{k \in \mathcal{K}(\mathcal{S},\bar{\mathcal{S}})} d^k$ where $\mathcal{K}(\mathcal{S},\bar{\mathcal{S}}) \subseteq \mathcal{K}$, be the set of commodities with their origin in $\mathcal{S}$ and their destination in $\bar{\mathcal{S}}$. $d_{(\mathcal{S},\bar{\mathcal{S}})}$ is then a lower bound on the amount of flow that must circulate across the cutset in any feasible solution. A set $\mathcal{C} \subseteq (\mathcal{S},\bar{\mathcal{S}})$ is a cover if the total capacity of the arcs in $(\mathcal{S},\bar{\mathcal{S}}) \backslash \mathcal{C}$ does not cover the demand, i.e. $\sum_{(i,j) \in (\mathcal{S},\bar{\mathcal{S}}) \backslash \mathcal{C}} u_{ij} \;<\; d_{(\mathcal{S},\bar{\mathcal{S}})}$. Moreover, the cover $\mathcal{C} \subseteq (\mathcal{S},\bar{\mathcal{S}})$ is minimal if it is sufficient to open any arc in $\mathcal{C}$ to cover the demand. For every cover $\mathcal{C} \subseteq (\mathcal{S},\bar{\mathcal{S}})$, the cover inequality

$$\sum_{(i,j) \in \mathcal{C}} y_{ij} \;\geq\; 1 \tag{3.2}$$

is valid for the CMND. The basic idea of this inequality is that one has to open at least one arc from the set $\mathcal{C}$ in order to meet the demand. In addition, it has been proven (Balas, 1975; Wolsey, 1975) that if $\mathcal{C}$ is a minimal cover, applying a lifting procedure yields a stronger inequality.

Flow Pack Inequalities (FPI). For any $\mathcal{L} \subseteq \mathcal{K}$ and cutset $(\mathcal{S},\bar{\mathcal{S}})$, let

$$x_{ij}^L = \sum_{k \in \mathcal{L}} x_{ij}^k, \;\; b_{ij}^L = \min\{u_{ij}, \sum_{k \in \mathcal{L}} d^k\}, \;\; \text{and} \;\; d_{(\mathcal{S},\bar{\mathcal{S}})}^L = \sum_{k \in \mathcal{K}(\mathcal{S},\bar{\mathcal{S}}) \cap L} d^k.$$

A flow pack $(\mathcal{C}_1, \mathcal{C}_2)$ is defined by two sets $\mathcal{C}_1 \subseteq (\mathcal{S},\bar{\mathcal{S}})$ and $\mathcal{C}_2 \subseteq (\bar{\mathcal{S}},\mathcal{S})$ such that $\mu = \sum_{(i,j) \in \mathcal{C}_1} b_{ij}^L - \sum_{(j,i) \in \mathcal{C}_2} b_{ji}^L - d_{(\mathcal{S},\bar{\mathcal{S}})}^L < 0$. Let $\mathcal{D}_1 \subset (\mathcal{S},\bar{\mathcal{S}}) \backslash \mathcal{C}_1$. The flow pack inequality is then defined as (Atamturk, 2001; Stallaert, 97)

$$\begin{aligned} \sum_{(i,j) \in \mathcal{C}_1} x_{ij}^L + \sum_{(i,j) \in \mathcal{D}_1} (x_{ij}^L - \min\{b_{ij}^L, -\mu\} y_{ij})) \;\;\leq\;\; & - \sum_{(j,i) \in \mathcal{C}_2} (b_{ji}^L + \mu)^+ (1 - y_{ji}) + \\ & \sum_{(j,i) \in (\bar{\mathcal{S}},\mathcal{S}) \backslash \mathcal{C}_2} x_{ji}^L + \sum_{(i,j) \in \mathcal{C}_1} b_{ij}^L. \end{aligned} \tag{3.3}$$

The cutting-plane procedure then mainly iterates on solving the linear relaxation LP and generating and adding to the LP formulation violated valid inequalities. It terminates when either the optimal solution is found, which is unlikely to happen very often, or when the improvement is smaller than ϵ. With $\bar{Z}$ and $(\bar{x}, \bar{y})$ standing for the optimal value and solution vector, respectively, of the LP with the currently generated VIs, Algorithm 2 displays the main steps of the procedure.

Algorithm 2 Lower-bound Computation

```
Z_Last ← 0; F ← 0^|A|; L ← 0^|A|; R ← 0^|A|
Solve the LP relaxation
while Z̄ − Z_Last > ε do
    Z_Last ← Z̄
    if ȳ is integer then
        return Z̄ and (x̄, ȳ)
    end if
    Generate and add violated SI
    Generate and add violated CI
    Update the L memory
    Generate and add violated FPI
    if New inequalities are added then
        Solve the LP relaxation
        Update Nb_LP
        Update the F and R memories
    end if
end while
return the lower bound Z̄, the corresponding solution (x̄, ȳ), and the memories F, L,
and R
```

Three different types of statistics characterizing attributes of LP solutions found while running the cutting plane algorithm are collected in three particular memories for subsequent utilization:

- $F \in \mathbb{N}^{|\mathcal{A}|}$, an $|\mathcal{A}|$-dimensional vector of *design-variable frequency*, representing how often an arc has been used in previous LP solutions. Set initially to the null vector, F is updated at each LP solution $(\bar{x}, \bar{y})$ by setting

$$F_{ij} = F_{ij} + 1 \text{ if } \bar{y}_{ij} > \beta, \ \forall (i,j) \in \mathcal{A},$$

  for a given threshold β indicating the importance of an arc (i,j) in the current LP solution.

- $L \in \mathbb{N}^{|\mathcal{A}|}$, an $|\mathcal{A}|$-dimensional vector of *violated cover inequality frequency*, representing how often design arcs were included in violated CI generated during the cutting-plane procedure. Similarly to F, initialized to the null vector, L is updated at each violated CI found

$$L_{ij} = L_{ij} + 1, \quad \forall (i,j) \in \mathcal{C},$$

where $\mathcal{C} \subseteq \mathcal{A}$ is the minimal cover obtained in the CI.

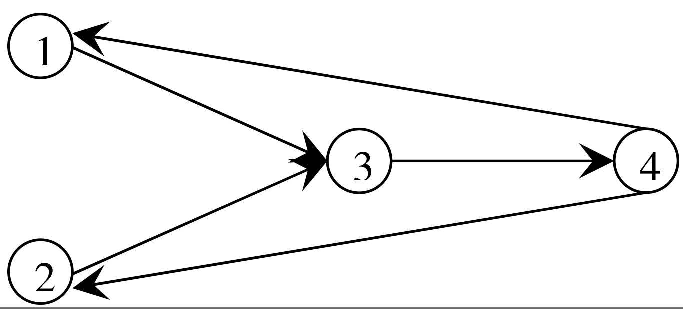
- $R \in \mathbb{R}^{|\mathcal{A}|}$, an $|A|$-dimensional vector of *accumulated reduced costs*. Similarly to the two previous memories, R is initialized to the null vector and is updated at each LP solution

  $$R_{ij} \;=\; R_{ij} + \bar{R}_{ij}, \quad \forall (i,j) \in \mathcal{A},$$

  where $\bar{R}$ is the reduced cost vector associated to the current LP optimal solution $(\bar{x}, \bar{y})$.

3.2 First feasible solution

Contrary to the CMND, it is not obvious to find a feasible initial solution for the DBCMND. The classical approach of first solving a linear relaxation and then rounding-up all design variables corresponding to used arcs in the LP solution (often used for the CMND) is not appropriate for the DBCMND. Indeed, except for some special cases, the integral solution obtained does not satisfy the design-balanced constrains. Figure 1 illustrates the infeasibility of such a rounding-up method for a small graph consisting of four nodes and five arcs. Notice that the LP solution satisfies the design-balanced requirements while the round-up solution does not (for nodes 3 and 4).



LP Solution		Round-up Solution
$y_{13} = 0.5$		$y_{13} = 1$
$y_{23} = 0.5$		$y_{23} = 1$
$y_{34} = 1$	Round Up $\Longrightarrow$	$y_{34} = 1$
$y_{41} = 0.5$		$y_{41} = 1$
$y_{42} = 0.5$		$y_{42} = 1$

Figure 1: Infeasibility of rounding-up LP solutions

To obtain a starting feasible solution, we propose to first reduce the size of the DBCMND, by closing a suitable subset of arcs, called $\widetilde{\mathcal{A}}$ and, then, solve the resulting reduced DBCMND problem using any available exact MIP code. The challenge is in selecting the suitable subset of arcs to close. On the one hand, one desires to close a sufficiently high number to yield a reduced design problem "easy" to address with a good MIP code. On the other hand, closing too many arcs may yield an easily addressed problem but a network too small to carry all the demand and, thus, inappropriate for the task at hand.

To efficiently address this challenge, we propose the α_0-*fixing* heuristic, which determines the suitable set $\widetilde{\mathcal{A}}$ using the compiled memories F, L, and R, introduced in Section 3.1. As illustrated in Algorithm 3, the heuristic starts with an empty set, and gradually adds arcs that are attractive given the information gathered while executing the cutting-plane algorithm, and that provide sufficient connectivity and capacity to the resulting network.

Algorithm 3 α_0-*Fixing Heuristic*

Require: F, L, R as given by the cutting-plane Algorithm 2

$\widetilde{A} = \emptyset$

Selection step based on F

Add to $\widetilde{\mathcal{A}}$ all arcs with frequency $\geq \alpha_0 Nb_{LP}$

Connectivity step based on $F + L$

Add arcs to $\widetilde{\mathcal{A}}$ to ensure each transshipment node has at least one incoming and one outgoing arc

Feasibility step based on R

Add arcs to $\widetilde{\mathcal{A}}$ to provide sufficient capacity for flows out and into demand and supply nodes, respectively.

Arcs are added in three consecutive steps. The *selection step* aims to select attractive arcs as defined by the frequency F of utilization in cutting-plane LP solutions. The idea is that arcs which are repeatedly used in optimal LP solutions are most likely to also be part of good, hopefully optimal, feasible solutions. Thus, given a threshold α_0, arc (i, j) is added to $\widetilde{\mathcal{A}}$ if $F_{ij} \geq \alpha_0 Nb_{LP}$, where Nb_{LP} is the number of LPs solved by the cutting-plane.

The *connectivity step* aims to provide the means for commodities to pass through each selected transshipment node in the network. This means that each transshipment node already in $\widetilde{\mathcal{A}}$, i.e., with at least an incoming/outgoing arc open, must have at least an outgoing/incoming arc open. Because the choice has to be made among arcs in $\mathcal{A} \backslash \widetilde{\mathcal{A}}$, which did not appeared often in cutting-plane LP solutions, we combine the measures of frequency F and the CI-frequency L. In fact, a frequent appearance of an arc in minimal covers of violated CIs means the arc has a good chance to be open and used in feasible solutions, and it is therefore a good candidate for the fixing heuristic. Consequently, for each node $i \in \mathcal{N}$ with at least one

- Incoming arc, $\sum_{j \in \widetilde{\mathcal{N}}_i^-} y_{ji} \geq 1$, and no outgoing arc, $\sum_{j \in \widetilde{\mathcal{N}}_i^+} y_{ij} = 0$, add to $\widetilde{\mathcal{A}}$ the arc such that $\text{argmax}_{j\in\mathcal{N}_i^+\setminus\widetilde{\mathcal{N}}_i^+}(F_{ij}+L_{ij})$;

- Outgoing arc, $\sum_{j\in\widetilde{\mathcal{N}}_i^+} y_{ij}\geq 1$, and no incoming arc, $\sum_{j\in\widetilde{\mathcal{N}}_i^-} y_{ji}=0$, add to $\widetilde{\mathcal{A}}$ the arc such that $\text{argmax}_{j\in\mathcal{N}_i^-\setminus\widetilde{\mathcal{N}}_i^-}(F_{ji}+L_{ji})$.

Finally, the feasibility step opens enough arcs at origins and destinations to provide sufficient capacity to satisfy the demand requirements. Because the previous step used the L memory, we aim for a certain degree of diversification in our selection and, thus, we use information based on the reduced costs R in this step. Therefore, for any supply or demand node such that

$$\sum_{j\in\widetilde{\mathcal{N}}_i^+} u_{ij} < w^k,\ i=O(k),\ \forall k\in\mathcal{K} \quad \Rightarrow \text{Add the arc such that } \text{argmin}_{j\in\mathcal{N}_i^+\setminus\widetilde{\mathcal{N}}_i^+}(R_{ij})$$
$$\sum_{j\in\widetilde{\mathcal{N}}_i^-} u_{ji} < w^k,\ i=D(k),\ \forall k\in\mathcal{K} \quad \Rightarrow \text{Add the arc such that } \text{argmin}_{j\in\mathcal{N}_i^-\setminus\widetilde{\mathcal{N}}_i^-}(R_{ji})$$

Once the set $\widetilde{\mathcal{A}}$ is determined using the α_0-fixing heuristic, we consider the restriction DBCMND$_{\widetilde{\mathcal{A}}}$ where all arcs in $\widetilde{\mathcal{A}}$ are free and all arcs in $\mathcal{A}\setminus\widetilde{\mathcal{A}}$ are closed. DBCMND$_{\widetilde{\mathcal{A}}}$ is then solved using a MIP code until optimality or a time or node limit is reached, the latter limits aiming to achieve the goal of as good a feasible solution as possible in a quick time. If the reduced problem is feasible, we proceed with the tabu search starting from this initial feasible solution. Otherwise, we decrease the value of α_0 and repeat the α_0-fixing heuristic to find a new and larger set $\widetilde{\mathcal{A}}$. We iterate on decreasing the value of α_0 and on performing α_0-fixing heuristic until a feasible solution is found.

According to our computational results, this heuristic has proved effective in identifying rapidly high-quality feasible solutions already competitive with those found by the best heuristic approaches available in the literature.

3.3 Cycle-based Tabu Search

The local search component of TS is used to explore the feasible design-variable search space of the DBCMND problem, starting from the feasible solution produced by the cutting plane and the α_0-fixing heuristic. The procedure follows the standard tabu-search template as shown in Algorithm 4. It is based on a large neighbourhood defined by closing and opening sets of arcs along cycles corresponding to arcs open in the current solution that are candidates to be closed. The neighbourhood and the corresponding *Arc-Balanced Cycle* procedure, preserving the feasibility of the solutions, are detailed in Section 3.3.1.

Algorithm 4 TS Local Search Heuristic:

Require: *Best-Solution* and *Current-Solution*
Initialization
Set *BestTS-Solution* and *Current-Design* to the *Current-Solution*
$F^{\text{TS}} = 0$ and $Nb_{\text{TS}} = 0$

Neighbourhood Exploration
Set up the *candidate list* of non-tabu arcs to drop
Determine the *neighbour list* by performing the *Arc-Balanced Cycle* procedure (Algorithm 5) for each candidate in the candidate list
for each neighbour in the neighbour list **do**
Solve the associated CMCF
If the CMCF is not feasible eliminate the neighbour and continue
end for
Select and move to the best neighbour
Update the tabu list, Nb_{TS}, *Current-Design*, *Best-Solution*, and *BestTS-Solution*, as well as the frequency memory F^{TS} with respect to *Current-Design*
if Computational time $>$ *TimeLimit* or *BestTS-Solution* did not improve for *NbIterTS* iterations **then**
return *Best-Solution*
else
Continue Neighbourhood Exploration
end if

Neighbours are evaluated by solving $\text{CMCF}_{\widetilde{y}}$, the minimum cost network flow problem associated to the design vector $\widetilde{y}$ of the current DBCMND feasible solution:

$$\begin{aligned}
\min_x \quad & \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\widetilde{\mathcal{A}}} c_{ij}^k x_{ij}^k \\
& \sum_{j\in\widetilde{\mathcal{N}}_i^+} x_{ij}^k - \sum_{j\in\widetilde{\mathcal{N}}_i^-} x_{ji}^k = d^k \quad \forall\, i \in \mathcal{N}, \forall\, k \in \mathcal{K}, \\
& \sum_{k\in\mathcal{K}} x_{ij}^k \le u_{ij}, \quad \forall\, (i,j) \in \widetilde{\mathcal{A}}, \\
& x_{ij}^k \ge 0, \quad \forall\, (i,j) \in \widetilde{\mathcal{A}},\ \forall\, k \in \mathcal{K},
\end{aligned}$$

where $\widetilde{\mathcal{A}} = \{(i,j) : \widetilde{y}_{ij} = 1\}$, $\widetilde{\mathcal{N}}_i^- = \{j \in \mathcal{N}_i^- : (j,i) \in \widetilde{\mathcal{A}}\}$, and $\widetilde{\mathcal{N}}_i^+ = \{j \in \mathcal{N}_i^+ : (i,j) \in \widetilde{\mathcal{A}}\}$. As $\text{CMCF}_{\widetilde{y}}$ is a continuous linear problem, it can be solved using any commercial LP software. If $\text{CMCF}_{\widetilde{y}}$ is infeasible, i.e., the current balanced design is not able to route all commodities, the current neighbour is eliminated and the search proceeds with the next one. The best neighbour is chosen to be the next incumbent.

To avoid cycling, recent moves are included in the tabu list, with a random number of iterations forbidding a closed arc to be included in any move. Besides the tabu list and the total number of TS solutions computed in Nb_{TS}, a memory, F^{TS} of $|\mathcal{A}|$ dimension, is updated at each iteration to record the frequency of a design arc being open in the *Current-Design* TS solutions. F^{TS} thus indicates the potential of arcs as experienced in explored TS feasible solutions and is used to identify new feasible solutions in the intensification phase (Section 3.4).

The heuristic stops when no improvement in the best solution is observed for a prespecified number of iterations or the overall computational time exceeds a time limit.

3.3.1 Neighbourhood Exploration

Adding or dropping arcs is the most common idea for defining moves for arc-based network formulations. Single-arc moves are simple to define and identify but do not perform very well on capacitated multicommodity network design. Multiple-arcs add/drop moves, modifying the status of several arcs simultaneously, proved significantly more effective, particularly when the corresponding arcs belonged to a network structure, such as a path or a cycle (e.g., Crainic et al., 2000; Ghamlouche et al., 2003; Hewitt et al.; Pedersen et al., 2009).

We therefore introduce a neighbourhood defined by a new *arc-balanced cycle* move that, while inspired by the cycle-based move of Ghamlouche et al. (2003), preserves the feasibility of the design formulation with respect to the design-balanced constraint. This property makes the move appropriate not only for the problem at hand but, more generally, for path or cycle-based formulations.

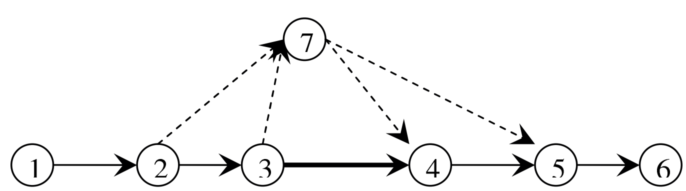


Figure 2: Add-drop move maintaining the design-balanced requirements

To define the new algorithm to generate arc-balanced cycle moves, observe that closing a given arc (r, t) introduces design unbalances at the origin and destination nodes of that arc. To maintain the design balance, it is necessary to introduce another path linking r and t. In general, the move involves closing a sub-path containing the arc (r, t) and opening a sub-path linking the two nodes so that the design-balanced constraints are maintained. Figure 2 illustrates this idea, where solid and dashed arcs are opened and closed, respectively. Closing arc $(3, 4)$ creates unbalances at nodes 3 and 4. Several possibilities exist to maintain the design-balanced requirements, e.g., $(3, 4)$ may be replaced by the sub-path $\{(3, 7), (7, 4)\}$, or the sub-path $\{(2, 3), (3, 4), (4, 5)\}$, containing $(3, 4)$, may be replaced by the sub-path $\{(2, 7), (7, 5)\}$. Notice that such a move does not account for the flow-distribution feasibility and the new design might not have sufficient capacity. Verifying this condition requires to solve the minimum cost network flow problem, which is performed in any case to evaluate the move. We therefore prefer to do it once and eliminate infeasible neighbours at the moment the potential move is evaluated.

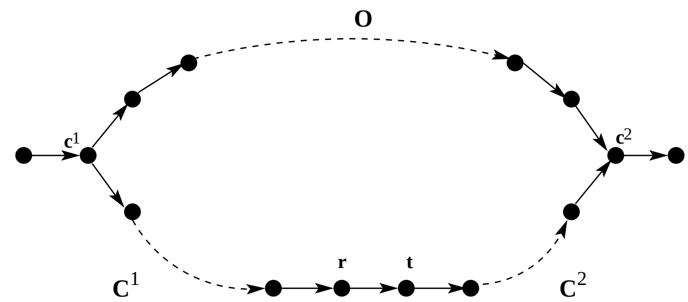


Figure 3: Special form of the arc-balanced cycle

Observe that any cycle composed of in and out sub-paths containing the arc to close so that, once the move is performed, the resulting design satisfies the design-balanced requirements, must satisfy what we call the *three-phase* property, that is, it must be composed of three sub-paths (C^1, $\{(r,t)\}$, C^2, O), illustrated in Figure 3, such that

$C^1 \subseteq \widetilde{\mathcal{A}}$ is a sub-path of open arcs between a given node c^1 and r;

$C^2 \subseteq \widetilde{\mathcal{A}}$ is a sub-path of open arcs between t and a given node c^2;

$O \subseteq \mathcal{A} \backslash \widetilde{\mathcal{A}}$ is a sub-path of closed arcs between c^1 and c^2.

The move is then performed by closing all arcs in $C^1 \cup \{(r,t)\} \cup C^2$ and opening the arcs in O. To increase the chance to identify flow-feasible arc-balanced cycles, we concentrate the search for C^1 and C^2 on arcs carrying the same subset of commodities as those moving on the arc to close (r,t). Such a selection of arcs also enables the generation of neighbours that are not too close to the current solution. Algorithm 5 describes the procedure identifying such an arc-balanced cycle move.

The algorithm proceeds in several steps. It first determines the subset of opened arcs that could be used in a cycle involving the arc to close a, by a labelling procedure starting from r and t and going backward and forward, respectively. The next step identifies the auxiliary network used to find the cycle for the possible move by 1) reversing all selected opened arcs and associating to them their negative fixed cost; and 2) adding all the closed arcs with their associated fixed cost. It is on the graph $\bar{G}_a$ that an appropriate shortest path from r to t is computed next, by a label-correcting procedure illustrated in Algorithm 6.

Algorithm 5 Arc-Balanced Cycle: Identify a cycle passing through arc $a = (r, t)$ and return subsets of arcs to drop and add while maintaining the design-balanced requirements

Require: $\widetilde{G} = (\widetilde{\mathcal{A}}, \widetilde{\mathcal{N}})$, the graph associated to the *Current-Design* $\widetilde{y}$
Require: $\widetilde{x}$, the current flow associated to *Current-Design*
Require: $\mathcal{K}^a = \{k \in \mathcal{K} | \widetilde{x}_a^k > 0\}$, the set of commodities moving on a at *Current-Design*
Initialization: $\mathit{ArcsToDrop} \leftarrow \emptyset$, $\mathit{ArcsToAdd} \leftarrow \emptyset$

{Determine the graph associated to arc a}
while not all $O(k), \forall k \in \mathcal{K}^a$, are marked **do**
  Mark all nodes and arcs starting from r and going backward
end while
while not all $D(k), \forall k \in \mathcal{K}^a$, are marked **do**
  Mark all nodes and arcs starting from t and going forward
end while
$G_a = (\mathcal{A}_a, \mathcal{N}_a)$ = the graph that includes all marked arcs and nodes
for each arc $a' \in G_a$ such that $\widetilde{x}_{a'}^k > 0, \forall k \in \mathcal{K}^a$ **do**
  Add its inverse to $\bar{G}_a$ with fixed cost $-f_{a'}$
end for
Remove from $\bar{G}_a$ all arcs and nodes that are not connected to r or to t
Add to $\bar{G}_a$ all arcs in $\mathcal{A} \backslash \widetilde{\mathcal{A}}$ touching nodes in $\bar{G}_a$, with their original fixed cost f

Compute shortest path from r to t in $\bar{G}_a$ by applying Algorithm 6
if no-path between r and t **then**
  STOP
else
  C_a is the cycle obtained by the shortest path and arc a
  $\mathit{ArcsToDrop} \leftarrow \{C_a \cap \widetilde{\mathcal{A}}\}$
  $\mathit{ArcsToAdd} \leftarrow \{C_a \cap \mathcal{A} \backslash \widetilde{\mathcal{A}}\}$
  Return $\mathit{ArcsToDrop}$, $\mathit{ArcsToAdd}$
end if

The goal of Algorithm 6 is to find a shortest path such that the induced cycle satisfies the three-phase property while avoiding negative cycles. A list of marked nodes is therefore built to avoid visiting a node more than once, and node labels (with values from 1 to 3) are used to track the status of the cycle and thus prohibit the generation of a negative one. The algorithm returns the value of the shortest path, $Dist[t]$, or indicates that such a path does not exist (distance equals ∞), in which case the arc (r, t) cannot be closed at that moment.

Algorithm 6 ThreePhasesSP: Compute shortest path from r to t such that the three-phase property is satisfied

---

Initialization
  $MarkedList \leftarrow \{r\}$
  $Dist[r] = 0, Label[r] = 1$
  $Dist[i] = \infty, Label[i] = 0, \ \forall i \in \widetilde{\mathcal{N}} \backslash r$
while $MarkedList \neq \emptyset$ **do**
  $n \leftarrow \text{argmin}_{i \in MarkedList} Dist[i]$, remove n from $MarkedList$
  if $n = t$ **then**
    Return $Dist[t]$
  else
    for $\forall i \in \widetilde{\mathcal{N}}_n^+$ such that $Label[n] < 4$ and $\bar{f}_{ni} \leq 0$ **do**
      if $(Dist[n] + \bar{f}_{ni} < Dist[i])$ **then**
        Update $Dist[i]$
        Add i to $MarkedList$
          If $(Label[n] = 1)$ and $(\bar{f}_{ni} < 0 \Rightarrow Label[i] = 1$
          If $(Label[n] = 1)$ and $(\bar{f}_{ni} \geq 0) \Rightarrow Label[i] = 2$
          If $(Label[n] = 2)$ and $(\bar{f}_{ni} \geq 0) \Rightarrow Label[i] = 2$
          If $(Label[n] = 2)$ and $(\bar{f}_{ni} < 0) \Rightarrow Label[i] = 3$
          If $(Label[n] = 3)$ and $(\bar{f}_{ni} < 0) \Rightarrow Label[i] = 3$
      end if
    end for
  end if
end while

Figure 4 illustrates the arc-balanced cycle algorithm. The current solution appears in Part a), solid and dashed lines standing for open and closed arcs, respectively. Arc $(4, 6)$, moving flow for commodities (O_1, D_1) and (O_2, D_2), is to be removed. Part b) shows the intermediate network obtained at the end of Step 2 and containing the opened arcs that could be included in paths passing through $(4, 6)$. Part c) shows the network obtained at the end of Step 3 indicating (solid lines), in opposite direction with negative fixed cost, the currently open arcs carrying the same subset of commodities as $(4, 6)$. Dashed lines represent closed arcs with their fixed costs. Part d)displays the neighbour design after the path $\{(2, 4), (4, 6)\}$ has been replaced by the path $\{(2, 3), (3, 6)\}$, both obtained using Algorithm 6 on the network of Part c). The design illustrated in Part d) satisfies the design-balanced requirements.

3.3.2 Candidate List

The neighbourhood introduced above may be very large, dimensions increasing rapidly with the size of the network. On one hand, the number of open arcs, possible candidates to be closed, could be large and, on the other hand, for each such arc, the number of possible cycles may be large as well. Moreover, a CMCF problem is solved to evaluate each contemplated neighbour and verify its flow-distribution feasibility. Exploring the entire neighbourhood could therefore rapidly become prohibitive.

We therefore select a restricted set of $\nu^1+\nu^2+\nu^3$ promising candidates, i.e., "expensive" arcs in terms of the quantity of flow moved relative to the cost of including and using the arc in the design. The set is built applying sequentially the following criteria:

- **Fixed cost**. We aim for a good balance between the fixed cost of the arc and the amount of flow it carries and define the $|\mathcal{A}|$-dimensional vector

$$\xi^1_{ij} = \begin{cases} \frac{f_{ij}}{\sum_k \widetilde{x}^k_{ij}}, & \text{if } \sum_k \widetilde{x}^k_{ij} > 0 \\ 0, & \text{otherwise.} \end{cases}$$

  A large ξ^1 value indicates a low usage of the arc given the fixed cost paid. We therefore sort the arcs in descending order of ξ^1 and add the first ν^1 to the candidate list.

- **Capacity**. We desire a good level of used capacity with respect to the flow moving on the arc, and define the $|\mathcal{A}|$-dimensional vector

$$\xi^2_{ij} = \begin{cases} \frac{\sum_k \widetilde{x}^k_{ij}}{u_{ij}}, & \text{if } \sum_k \widetilde{x}^k_{ij} > 0 \\ 0, & \text{otherwise.} \end{cases}$$

  The most promising candidates are those showing small ξ^2 values indicating that little of the arc capacity is used. Thus, we sort arcs in ascending order of ξ^2 and select the first ν^2 arcs.

- **Unit price**. We aim for a low level of *effective* unit price for the flows on the arcs, given the fixed cost of the arcs. Define the $|\mathcal{A}|$-dimensional vector

$$\xi^3_{ij} = \begin{cases} \frac{f_{ij}+\sum_k c^k_{ij}\widetilde{x}^k_{ij}}{u_{ij}}, & \text{if } \sum_k \widetilde{x}^k_{ij} > 0 \\ 0, & \text{otherwise.} \end{cases}$$

  Similarly to the first criterion, large ξ^3 values indicate arcs where we pay a high total cost for the flow moved relative to their capacity. We thus sort arcs in descending order of ξ^3 and select the first ν^3 arcs.

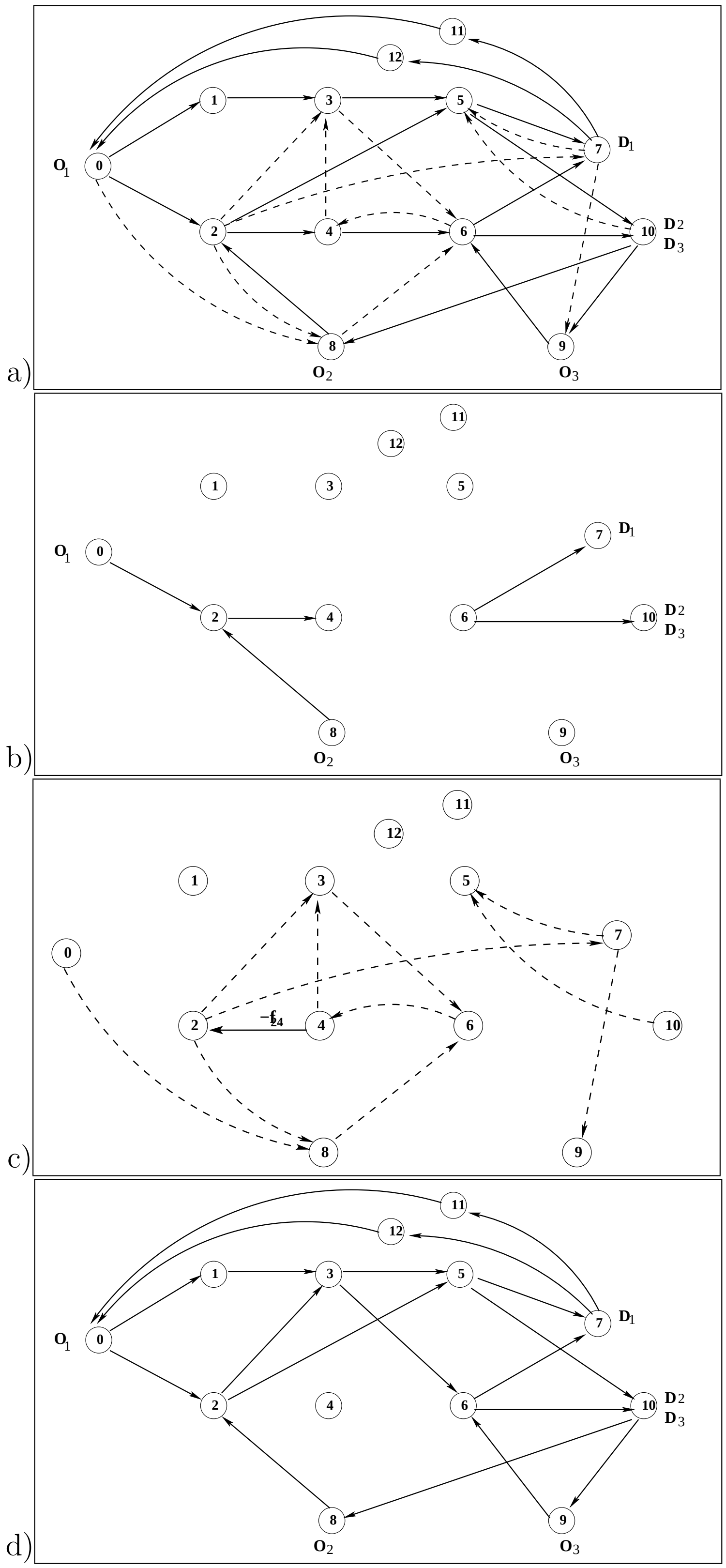


Figure 4: Illustration of a cycle move identification

3.4 TS intensification - New feasible solution

The definition of the intensification phase of the TS algorithm is also motivated by the idea of combining exact and heuristic methods. We desire to intensify the search around a set of *good attributes* identified during the previous neighbourhood-exploration phase, reflected in the frequency memory F^{TS} compiled by Algorithm 4, as well as during the initial lower-bound computation by the cutting-plane Algorithm 2.

This set is used to reduce the size of the CMND problem, which is then solved by using a MIP exact algorithm. The reduction is obtained by fixing to 1 (opening) a number of arcs in the current network displaying the good attributes, and keeping all the remaining arcs free for the MIP method. The solution of the reduced CMND becomes then the starting point of the next neighbourhood-exploration phase.

Algorithm 7 illustrates the main steps of this α_1-*Fixing* heuristic. The procedure is similar to the α_0-fixing heuristic of Section 3.2, except for the utilization of the frequency memory F^{TS} instead of F.

Algorithm 7 α_1-*Fixing Heuristic*

Require: L, R compiled by the cutting-plane Algorithm 2
Require: F^{TS} compiled by the TS Algorithm 4
1: $\widetilde{A} = \emptyset$
2: *Selection step* based on F^{TS}
3: Add to $\widetilde{\mathcal{A}}$ all arcs with frequency $\geq \alpha_1 Nb_{\text{TS}}$
4: *Connectivity step* based on $F^{\text{TS}} + L$
5: Add arcs to $\widetilde{\mathcal{A}}$ to ensure each transshipment node has at least one incoming and one outgoing arc
6: *Feasibility step* based on R
7: Add arcs to $\widetilde{\mathcal{A}}$ to provide sufficient capacity for flows out and into demand and supply nodes, respectively.

4 Computational results

The objectives of the computational experiments are threefold: 1) to test the effectiveness of the cutting-plane procedure, developed originally for the general network design problem, in the context of DBCMND; 2) to evaluate the quality of the initial feasible solutions obtained by the α_0-*fixing* heuristic; and 3) to evaluate the performance of the MIP-TS matheuristic and its capability to identify high-quality feasible solutions compared to the best existing solution methods. We actually compare our results with those obtained by the meta-heuristic of Pedersen et al. (2009), as well as to the best solution obtained by the Branch-and-Cut method (B&C) of CPLEX (version 12). A computational time limit of one hour was imposed to all methods.

The procedures were coded in C++. The LP relaxations within the cutting-plane procedure were solved to optimality using the option *Dualopt* of CPLEX (version 12). All the restrictions DBCMND$_{\widetilde{\mathcal{A}}}$ were solved using the B&C of CPLEX with a time limit of one hour and a node limit of 100. Experiments were performed on a network of Dual-Core AMD Opteron (using a single thread) workstations with 8 Gigabytes of RAM operating under SunOS 5.1.

The performance of the proposed matheuristic is evaluated on a set of network design instances with various characteristics used in several papers (Ghamlouche et al., 2003; Pedersen et al., 2009; Chouman et al., 2011) and described in Crainic et al. (2001). These problem instances, identified as Sets **C** and **R**, consist of general transshipment networks with one commodity per origin-destination pair and no parallel arcs. Positive transportation cost, fixed cost, and capacity are associated with each arc. Note that the transportation costs on any given arc are the same for all commodities.

Set **C** consists of 43 instances characterized by their number of nodes, arcs, and commodities, noted $|N|, |A|$, and $|K|$, respectively. Two additional letters are used to characterize the fixed cost level, "F" for high and "V" for low, relatively to the transportation cost, and the capacity level, "T" for tight and "L" for loose, compared to the total demand. The set of instances **R** consists of 81 problems, nine sets of nine instances each. Each set is characterized by the same number of nodes, arcs, and commodities, instances displaying various levels of fixed cost and capacity ratios. Thus, "F01" for low, "F05" for medium, and "F10" for high, are used to qualify the importance of the fixed cost with respect to the transportation cost, while "C1" for loose, "C2" for medium, and "C8" for tight, to qualify the tightness of the capacity compared to the total demand. To facilitate comparisons, we present the results for the 24 **C** instances and the 54 **R** instances used in Pedersen et al. (2009). These are medium to large-size instances with various levels of cost and capacity ratios.

Our primary measure of performance is the gap between the reference solution z^*, which is either the overall best feasible solution or the initial solution obtained by the α_0-*Fixing* procedure, and a given solution z computed as:

$$\Delta z^*/z = \frac{100(z^* - z)}{z^*} \tag{4.1}$$

In order not to overload the paper, we report average results over the **C** and **R** instances. Detailed results for each instance are included in the Appendix. The next subsection addresses the first two objectives stated above, while the following subsection is dedicated to the third.

4.1 Evaluation of the lower bound and first feasible solution procedures

The scope of this section is to analyze the performance of the cutting-plane method in the context of the DBCMND. We aim to examine, in particular, the effectiveness of this algorithm in improving the lower bound of the DBCMND while compiling characteristics and attributes of good solutions. We also aim to evaluate the performance of the α_0-*fixing* in identifying high-quality initial feasible solutions based on the informations compiled during the computations of the cutting-plane algorithm.

Table 1 shows the results obtained by the cutting-plane algorithm for the **C** and **R** sets averaged according to the problem dimensions in terms of numbers of nodes, arcs, and commodities. For each such group of instances (Column Description), the table indicates the number of instances in the group (Column Nb), the gap between the lower bound obtained by the cutting-plane and the first LP bound (Column GapLP), the total number of cuts generated in the cutting-plane (Column Cuts), and the total number of LP solved (Column NbLP).

Set **C**				
Description	Nb	GapLP	Cuts	NbLP
20,230,200	(4)	30.85%	3148	20
20,300,200	(4)	21.59%	2370	22
30,520,100	(4)	21.10%	2003	23
30,520,400	(4)	16.18%	3596	13
30,700,100	(4)	18.99%	1770	30
30,700,400	(4)	18.70%	3953	12
Average	(24)	21.23%	2806	20
Set **R**				
Description	Nb	GapLP	Cuts	NbLP
20,220,40	(9)	38.87%	1107	25
20,220,100	(9)	33.60%	1693	29
20,220,200	(9)	28.87%	2072	24
20,320,40	(9)	45.04%	1888	33
20,320,100	(9)	39.62%	2616	21
20,320,200	(9)	34.66%	3287	19
Average	(54)	36.78%	2111	25

Table 1: Evaluation of the Cutting-Plane Algorithm

The results show clearly the effectiveness of the cutting-plane algorithm in improving the quality of the LP bounds. The overall averaged gap improvement for the **C** instances reaches 21.23%, while it is 36.78% for the **R** instance set. Although the average numbers of cuts generated and LP solved may seem relatively high, the associated computational effort is low. We actually observe that very short solving times for the first LP, with averages of 5.75 and 2.45 CPU seconds on average for the **C** and **R** sets, respectively. We also observe that the time required for the cut generations is almost negligible, while solving the LP relaxation after each round of cut generation is more efficient than for the first LP, because it consists in re-optimizing from a previous optimal basis (the simplex method of CPLEX is applied with the *Dualopt* option).

These results support our claim that the cutting-plane algorithm is effective in improving the bounds within a short computing effort, even when repetitively solving different and LP models: 20 and 25 on average for the **C** and **R** instance sets, respectively. As discussed in Section 3.2, these multiple solutions of LP models provide the means to compile the memories that are then used in the α_0-*fixing* heuristic to guide the search towards good initial feasible solutions.

The value of the parameters in the implementation of the α_0-*fixing* procedure are $\alpha_0 = 0.45$ and $\beta = 0.3$. These values were selected based on computational experiments where, with the objective of including a suitable number of arcs in the network, the median values of the F and L memories were first computed, then several values around these medians were tested.

Table 2 displays the results for all **C** and **R** instances averaged according to problem dimensions. In addition to the Description and Nb columns, Columns FS/TS and FS/CPLEX display the gaps between the initial feasible solution, FS, found by the α_0-*fixing* heuristic and the solutions obtained by the Tabu Search of Pedersen et al. (2009) and CPLEX, respectively. Columns FS/LB_CPLEX and FS/LB indicate the gap of the initial solution FS with respect to the best lower bound found by the B&C of CPLEX and the lower bound found by the cutting-plane, respectively.

The negative values in the FS/TS column indicate that the α_0-*fixing* heuristic outperforms the current best heuristic for the the DBCMND, namely the Tabu Search method of Pedersen et al. (2009). The results also show that the improvement is more important for difficult problems characterized by large number of commodities. Moreover, the proposed heuristic is also competitive in solution quality with the best feasible solution found by CPLEX after one hour of CPU time, with overall differences as low as 0.33% and 0.54% for the **C** and **R** sets, respectively.

To sum up these comparative results, the proposed α_0-*fixing* heuristic yields initial feasible solutions that improve over the solutions obtained by the Tabu Search procedure of Pedersen et al. (2009) and the state-of-the-art B&C of CPLEX for 22 and 8 of the 24 instances of Set **C**, respectively. The figures for the 54 instances of Set **R** are 46 and 11, respectively. This performance appears even more remarkable when comparing the computing efforts (one hour was given to CPLEX).

Set **C**					
DESCRIPTION	NB	FS/TS	FS/CPLEX	FS/LB_CPLEX	FS/LB
20,230,200	(4)	-4.08%	0.66%	4.80%	5.12%
20,300,200	(4)	-3.37%	1.58%	4.19%	4.57%
30,520,100	(4)	0.00%	2.87%	6.05%	6.20%
30,520,400	(4)	-4.01%	-0.11%	2.52%	1.98%
30,700,100	(4)	-1.66%	1.29%	2.96%	3.25%
30,700,400	(4)	-7.32%	-1.11%	2.63%	2.60%
Average	(24)	-3.27%	0.33%	5.13%	5.51%
Set **R**					
DESCRIPTION	NB	FS/TS	FS/CPLEX	FS/LB_CPLEX	FS/LB
20,220,40	(9)	-1.32%	3.19%	4.25%	7.27%
20,220,100	(9)	-3.26%	1.22%	4.82%	5.51%
20,220,200	(9)	-5.51%	-0.37%	2.70%	2.91%
20,320,40	(9)	-1.37%	3.98%	6.69%	8.45%
20,320,100	(9)	-3.00%	1.61%	6.91%	7.36%
20,320,200	(9)	-4.87%	-6.40%	4.73%	4.34%
Average	(54)	-3.22%	0.54%	5.02%	5.97%

Table 2: Initial solution comparisons

4.2 Evaluation of the MIP-TS matheuristic

We present comparative results for the MIP-TS matheuristic, the Tabu Search metaheuristic of Pedersen et al. (2009), and the B&C of CPLEX, version 12.

The value of the parameters used in the implementation of the proposed MIP-TS matheiristic are $\alpha_1 = 0.85$, $NbIterTS = 10$, $NbIterProcess = 10$, and $\nu^1 = \nu^2 = \nu^3 = 7$. The α_1 value was selected with the goal of fixing about a third of the arcs in the α_1-fixing heuristic. A preliminary computational study targeted ν^1, ν^2, and ν^3, and several values and combinations thereof, ranging from 0 to 20, were examined. Several value combinations performed equally well, the value 7 being representative of this lot.

A random factor is associated to the tabu tenure of closed arcs in the TS metaheuristic (Algorithm 4), generated between $[1, 5]$ using the *srand()* function of C++. To account for this randomness and its impact on the stability of the proposed matheuristic, we run five repetitions for each instance (detailed results are presented in the Appendix). We observe a very high consistency of results and stability of the algorithm. The average standard deviation for the **C** instances is 0.03%, 18 instances out of the 24 displaying a standard deviation equal to 0. The figures for the instances of Set **R** are 0.04% and 32 out of 54, respectively. Based on this consistency and accuracy, we use the average solution, noted BS, in all the tables and comparisons that follow.

Obviously, we are expecting that the proposed MIP-TS algorithm outperforms the TS of Pedersen et al. (2009) and the B&C of CPLEX because of the performance of the initial feasible solution discussed in the previous subsection. Consequently, the aim here is to evaluate the performance of the other components of the method, namely its neighbourhood exploration and intensification phases, given the same time limit as the other two methods.

Table 3 displays the results for the **C** and **R** instances averaged according to problem dimensions. Columns Description and Nb have the same meaning as in previous tables. Columns BS/TS, BS/CPLEX, and BS/FS display the gaps between the average solution BS found by the proposed matheuristic and the solution obtained by Pedersen et al. (2009), CPLEX solution, and the initial-solution procedure, respectively. Columns BS/LB_CPLEX and BS/LB indicate the gaps BS with respect to the lower bound of the B&C of CPLEX after one hour of CPU time and the lower bound of the cutting-plane procedure, respectively.

Set **C**						
Description	Nb	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
20,230,200	(4)	-4.82%	-0.06%	-0.72%	4.11%	4.44%
20,300,200	(4)	-4.02%	0.97%	-0.63%	3.60%	3.97%
30,520,100	(4)	-1.80%	1.14%	-1.82%	4.40%	4.55%
30,520,400	(4)	-4.01%	-0.11%	0.00%	2.52%	1.98%
30,700,100	(4)	-2.33%	0.64%	-0.66%	2.33%	2.62%
30,700,400	(4)	-7.32%	-1.11%	0.00%	2.63%	2.60%
Average	(24)	-4.13%	-0.56%	-0.87%	4.34%	4.73%
Set **R**						
Description	Nb	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
20,220,40	(9)	-2.85%	1.71%	-1.56%	2.80%	5.90%
20,220,100	(9)	-3.46%	1.03%	-0.19%	4.65%	5.33%
20,220,200	(9)	-5.59%	-0.45%	-0.08%	2.62%	2.83%
20,320,40	(9)	-3.46%	1.96%	-2.23%	4.70%	6.49%
20,320,100	(9)	-3.86%	0.81%	-0.82%	6.16%	6.61%
20,320,200	(9)	-4.97%	-6.49%	-0.10%	4.64%	4.24%
Average	(54)	-4.03%	-0.24%	-0.83%	4.26%	5.23%

Table 3: Comparative results aggregated with respect to problem dimension

The results indicate clearly the superiority of the proposed MIP-TS matheuristic, relative to all the other solutions, in identifying high-quality feasible solutions. It significantly outperforms the meta-heuristic of Pedersen et al. (2009), improving the results for all **C** instances and for 50 out of the 54 **R** instances. Moreover, the improvement gap increases for difficult problems characterized by large number of commodities.

The proposed method also compares very well with B&C of CPLEX. It identifies better solutions for 9 of the 24 **C** instances and for 14 of the 54 **R** instances. It is very competitive for instances characterized by small to medium number of commodities, while outperforming CPLEX when the number of commodities increases. Indeed, CPLEX cannot find feasible solutions for 4 out of the 8 instances with 400 commodities, while the method we propose provides solutions with gaps ranging from 2.28% to 4.02%. Notice that only instances fro which CPLEX found a feasible solution are included in the figures of Column BS/CPLEX, which underestimates the performance of the proposed matheuristic. We therefore expect this new method to be the best when addressing real-world problems (within the dimensions experimented with, of course).

The importance of the neighbourhood exploration and intensification phases is reflected in the negative figures of Column BS/FS indicating the improvement these two phases brought to BS over the initial solution FS. More in detail, the initial solution is improved for 11 out of the 24 **C** instances, where 8 were identified by the intensification phase and 3 through the exploration phase. For the 54 **R** instances, initial solutions were improved for 31 instances, 16 achieved by the intensification phase and 15 through the exploration phase. Notice that the 0% improvement for the instances with 400 commodities is due to the limited time remaining, if any, for these phase after running the α_0-*fixing* heuristic.

Set **C**							
Description		Nb	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
V	L	(6)	-3.17%	0.18%	-0.63%	2.58%	2.91%
V	T	(6)	-3.03%	0.15%	-0.30%	1.49%	1.73%
F	L	(6)	-6.16%	1.09%	-0.83%	5.14%	4.90%
F	T	(6)	-3.82%	0.49%	-0.79%	3.84%	3.90%
Average		(24)	-4.13%	-0.56%	-0.87%	4.34%	4.73%

Set **R**							
Description		Nb	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
	C1	(6)	-1.36%	0.48%	-0.09%	0.79%	1.37%
F01	C2	(6)	-1.95%	0.53%	-1.94%	0.99%	1.59%
	C8	(6)	-2.61%	0.01%	-0.49%	1.61%	2.62%
	C1	(6)	-1.84%	0.73%	-2.36%	5.81%	7.39%
F05	C2	(6)	-3.80%	-1.53%	-0.38%	4.69%	6.14%
	C8	(6)	-6.30%	-0.64%	-0.20%	3.15%	3.88%
	C1	(6)	-2.91%	0.63%	-0.63%	8.99%	10.61%
F10	C2	(6)	-5.93%	-1.68%	-0.79%	7.13%	7.98%
	C8	(6)	-9.55%	-0.69%	-0.60%	5.20%	5.51%
Average		(54)	-4.03%	-0.24%	-0.83%	4.26%	5.23%

Table 4: Comparative results aggregated with respect to fixed cost and capacity ratios

Table 4 displays the same results averaged according to the different levels of fixed cost and capacity ratios. These figures indicate that the performance of the proposed MIP-TS method increases with the problem difficulty. Indeed, he gaps of the best solution computed with respect to the lower bounds provided by CPLEX and the cutting-plane procedure (last two columns) are smaller when, for the same level of cost ratio, the level of the capacity ratios increases from loose (C1) to tight (C8). We take these results to support the claim that the proposed MIP-TS matheuristic is suitable for hard real-world problems characterized by large size and limited resources.

5 Conclusions and Perspectives

We introduced a new MIP-TS matheuristic for the design-balanced capacitated multicommodity network design, one of the premier formulations for the service network design problem with asset management concerns increasingly faced by carriers within their tactical planning processesd.

The matheuristic combines a cutting-plane procedure efficiently computing tight lower bounds and a Tabu Search meta-heuristic exploiting a new cycle-based neighbourhood satisfying the design-balanced requirements. Learning mechanisms embedded into each of these procedures help in fixing variables and identifying good starting and intensification solutions.

An extensive computational study first showed that the cutting-plane procedure, initially proposed for the fixed-charge, multicommodity capacitated network design problem is also very efficient for the special structure of DBCMND, cutset-based inequalities in particular.

This cutting-plane procedure, together with appropriate learning mechanisms and variable-fixing techniques, was also shown to yield an efficient algorithm to identify high-quality feasible solutions, which may then serve as the starting point for improvement meta-heuristics. This capability of the proposed method is by itself remarkable as identifying feasible solutions to the DBCMND was shown previously to be NP-Hard.

Numerical experiments also shown the efficiency of the proposed design-balanced cycle neighbourhood exploration and the intensification mechanisms in improving already good initial feasible solutions. These combined features provided the means for the proposed MIP-TS matheuristic to outperform existing solution methods in solution quality and computational effort. It currently stands as the best heuristic for the DBCMND.

The fundamental ideas on which the new matheuristic is built are general in nature and open interesting research perspectives in hybridizing mathematical programming and meta-heuristics for network design problems. We are currently following some of these avenues, including adapting these ideas for other hard transportation planning problems, such as the management of power-unit fleets (e.g., locomotives in rail transportation). We plan to report on these developments in the near future.

Acknowledgments

While working on this project, T.G. Crainic was the NSERC Industrial Research Chair on 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, while M. Chouman was postdoctoral scientist with the Chair.

The authors thank Ms. Geneviève Hernu, analyst with the Chair, for her much welcome assistance in conducting the experimental phase of this project, and the CIRRELT for access to the material and human resources of its computing network. Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grants programs, and by the partners of the Chair, CN, Rona, Alimentation Couche-Tard and the Ministry of Transportation of Quebec. The computing equipment principally used was acquired with the support of the Canadian Foundation for Innovation and its partners, mainly the Quebec Ministry of Education and UQAM.

References

J. Andersen, T.G. Crainic, and M. Christiansen. Service Network Design with Management and Coordination of Multiple Fleets. *European Journal of Operational Research*, 193(2):377–389, 2009a.

J. Andersen, T.G. Crainic, and M. Christiansen. Service Network Design with Asset Management: Formulations and Comparative Analyzes. *Transportation Research Part C: New Technologies*, 17(2):397–207, 2009b.

J. Andersen, M. Christiansen, T.G. Crainic, and R. Grønhaug. Branch-and-Price for Service Network Design with Asset Management Constraints. *Transportation Science*, 46(1):33–49, 2011.

A.P. Armacost, C. Barnhart, and K.A. Ware. Composite Variable Formulations for Express Shipment Service Network Design. *Transportation Science*, 36(1):1–20, 2002.

A. Atamturk. Flow pack facets of the single node fixed-charge flow polytope. *Operations Research Letters*, 29:107–114, 2001.

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

E. Balas. Facets of the knapsack polytope. *Mathematical Programming*, 8:146–164, 1975.

T. Bektas and T.G. Crainic. A brief overview of intermodal transportation. In G. Taylor, editor, *Logistics Engineering Handbook*, chapter 28, pages 1 – 16. Taylor and Francis Group, 2008.

M. Chouman, T.G. Crainic, and B. Gendron. A cutting-plane algorithm for multicommodity capacitated fixed charge network design. Technical report, Center for research on transportation, Université de Montréal, 2009.

M. Chouman, T.G. Crainic, and B. Gendron. Commodity Representations and Cutset-Based Inequalities for Multicommodity Capacitated Fixed Charge Network Design. Technical Report CIRRELT-2011-56, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada, 2011.

T.G. Crainic. Service network design in freight transportation. *European Journal of Operational Research*, 122:272–288, 2000.

T.G. Crainic. Long-Haul Freight Transportation. In Hall, R.W., editor, *Handbook of Transportation Science*, pages 451–516. Kluwer Academic Publishers, Norwell, MA, second edition, 2003.

T.G. Crainic and K. Kim. Intermodal Transportation. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, chapter 8, pages 467–537. North-Holland, Amsterdam, 2007.

T.G. Crainic, A. Frangioni, and B. Gendron. Multicommodity capacitated network design. In P. Soriano and B. Sanso, editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academics Publisher, 1999.

T.G. Crainic, M. Gendreau, and J. M. Farvolden. A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, 12:223–236, 2000.

T.G. Crainic, A. Frangioni, and B. Gendron. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112:73–99, 2001.

B. Gendron and T.G. Crainic. Relaxations for multicommodity capacitated network design problems. Technical report, Publication CRT-945, Centre de recherche sur les transports, Université de Montréal, 1994.

I. Ghamlouche, T.G. Crainic, and M. Gendreau. Cycle-based neighbourhoods for fixed charge capacitated multicommodity network design. *Operations Research*, 51:655–667, 2003.

F. Glover. Futur paths for integer programming and links to artificial intelligence. *Computer & Operations Research*, 1(13):533–549, 1986.

F. Glover and M. Laguna. In *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, 1997.

M. Hewitt, G. L. Nemhauser, and M. W. P. Savelsberg. Combining Exact and Heuristic Approaches for the Capacitated Fixed-Charge Network Flow Problem. *INFORMS Journal on Computing*, 22(2):314–325, YEAR =.

M.F. Lai and H.K. Lo. Ferry Service Network Design: Optimal Fleet Size, Routing and Scheduling. *Transportation Research Part A: Policy and Practice*, 38:305–328, 2004.

T. L. Magnanti and R. T. Wong. Network design and transportation planning: models and algorithms. *Transportation Science*, 18:1–55, 1984.

M. Minoux. Network synthesis and optimum network design problems: models, solution methods and applications. *Networks*, 19:313–360, 1989.

M.B. Pedersen, T.G. Crainic, and O.B.G. Madsen. Models and Tabu Search Metaheuristics for Service Network Design with Asset-Balance Requirements. *Transportation Science*, 43(2):158–177, 2009.

K.R. Smilowitz, A. Atamtürk, and C.F. Daganzo. Deferred Item and Vehicle Routing within Integrated Networks. *Transportation Research Part E: Logistics and Transportation*, 39:305–323, 2003.

J. Stallaert. The complementary class of generalized flow cover inequalities. *Discrete Applied Mathematics*, 77:73–80, 97.

L. Wolsey. Faces of linear inequalities in 0-1 variables. *Mathematical Programming*, 8: 165–178, 1975.

Appendix

Tables 5 and 6 display the detailed results for the instance of sets **C** and **R**, respectively. For each instance, the respective table gives its description, the best feasible solution obtained by the Tabu Search of Pedersen et al. (2009) (Column TS), the best feasible solution and the lower bound obtained by the B&C of CPLEX (Columns CPLEX and LB_CPLEX, respectively), the initial feasible solution obtained by the α_0-*fixing* heuristic (Column FS), the lower bound and the number of cuts generated by the cutting-plane algorithm (Columns LOB and CUTS, respectively), as well as, for the five repetitions of the proposed MIP-TS matheuristic, the average solution, best solution, and standard deviation (Columns AVERBS, BESTBS, and STDBS, respectively).

INSTANCE	TS	CPLEX		MIP-TS					
		CPLEX	LB_CPLEX	FS	LB	CUTS	AVERBS	BESTBS	STDBS
C20,230,200,V,L	102919	98512	94305	98421	93774	3261	98421	98421	0
C20,230,200,F,L	150764	140843	134258	141744	134043	3476	141744	141744	0
C20,230,200,V,T	103371	101089	98754	103103	98383	2756	101258	101244	23
C20,230,200,F,T	149942	142452	134792	142638	134420	3098	141130	141130	0
C20,300,200,V,L	82533	77570	75583	80831	75267	2226	78953	78576	299
C20,300,200,F,L	128757	119945	115391	121302	114762	2816	121106	121106	0
C20,300,200,V,T	78571	76350	75832	76545	75507	2090	76545	76545	0
C20,300,200,F,T	116338	112358	108338	113412	108154	2348	113412	113412	0
C30,520,100,V,L	55981	54810	54096	55894	53909	1553	55268	55159	75
C30,520,100,F,L	104533	99717	93542	106078	93620	3181	101603	101129	265
C30,520,100,V,T	54493	53034	52734	53224	52493	1048	53224	53224	0
C30,520,100,F,T	105167	102919	97576	106324	97684	2231	104491	104426	89
C30,520,400,V,L	119735	115487	113313	115477	113295	3259	115477	115477	0
C30,520,400,F,L	162360	na	146508	153943	149754	4199	153943	153943	0
C30,520,400,V,T	120421	117214	115741	116959	115738	2909	116959	116959	0
C30,520,400,F,T	161978	na	152245	155863	152317	4016	155863	155863	0
C30,700,100,V,L	49429	48693	48538	49268	48187	1428	49139	49139	0
C30,700,100,F,L	63889	61430	59512	62267	59442	2523	62000	62000	0
C30,700,100,V,T	48202	46750	46224	46878	46101	1450	46875	46865	6
C30,700,100,F,T	58204	56337	55085	57701	55035	1677	56599	56599	0
C30,700,400,V,L	103932	101866	97809	99588	97814	4148	99588	99588	0
C30,700,400,F,L	157043	na	133424	139088	133503	4856	139088	139088	0
C30,700,400,V,T	103085	97838	95784	97901	95807	3152	97901	97901	0
C30,700,400,F,T	141917	na	129688	132999	129717	3654	132999	132999	0

Table 5: Results for the 24 instances of Set **C**

Instance	TS	CPLEX		MIP-TS					
		CPLEX	LB_CPLEX	FS	LB	Cuts	AverBS	BestBS	StdBS
r13,F01,C1	147837	147349	147349	148494	147089	641	148494	148494	0
r13,F05,C1	281668	277944	273851	298494	260671	1838	282843	281087	1603
r13,F10,C1	404434	385396	385358	417877	358674	2368	403596	403596	0
r13,F01,C2	159852	155887	155881	155887	154156	642	155887	155887	0
r13,F05,C2	311209	295180	295152	303859	280920	1172	302536	301729	737
r13,F10,C2	470034	434383	413509	454625	397156	1573	442505	442410	130
r13,F01,C8	225339	218787	218765	224632	213542	292	221344	219975	1254
r13,F05,C8	512027	491804	481358	497877	471641	603	497435	497325	247
r13,F10,C8	875984	782049	770345	798947	749648	835	796903	792096	2891
r14,F01,C1	431562	422709	422667	423538	418358	1313	423538	423538	0
r14,F05,C1	811102	790716	754449	808897	744067	2727	801872	797767	2295
r14,F10,C1	1193950	1145783	1043786	1210290	1032480	3514	1207090	1207090	0
r14,F01,C2	465762	452591	452546	455054	449318	1036	455054	455054	0
r14,F05,C2	942678	884673	851889	890673	845183	1734	890673	890673	0
r14,F10,C2	1401880	1317261	1228554	1308890	1234430	2312	1308890	1308890	0
r14,F01,C8	720882	702781	698606	706661	692517	572	706661	706661	0
r14,F05,C8	1795650	1695949	1642525	1708510	1628010	1044	1708510	1708510	0
r14,F10,C8	2997290	2787042	2636190	2824950	2630360	982	2807808	2804980	6324
r15,F01,C1	1039440	1017740	1016255	1020910	1008050	1778	1020910	1020910	0
r15,F05,C1	2170310	2024138	1946953	2028140	1943490	3644	2023750	2023750	0
r15,F10,C1	3194270	3028908	2826043	3003990	2819330	4427	3003990	3003990	0
r15,F01,C2	1205790	1176047	1167059	1182430	1163080	1455	1177802	1176990	1047
r15,F05,C2	2698680	2681189	2467568	2581910	2476600	2497	2581910	2581910	0
r15,F10,C2	4447950	4125923	3816835	4126150	3840130	2700	4121320	4121320	0
r15,F01,C8	2472860	2401176	2391424	2404240	2379310	567	2403970	2403970	0
r15,F05,C8	6067350	5795320	5794821	5797170	5767040	780	5797170	5797170	0
r15,F10,C8	10263600	9105014	9104291	9115830	9082920	804	9115830	9115830	0
r16,F01,C1	142692	140082	140077	141172	139829	1039	141095	140787	172
r16,F05,C1	261775	251554	244299	277712	237076	3110	261049	261049	0
r16,F10,C1	374819	348805	325972	349476	320035	3712	349476	349476	0
r16,F01,C2	145266	142381	142367	159168	141588	1001	143840	143689	161
r16,F05,C2	277307	259639	255356	278575	246361	2355	274751	271795	1747
r16,F10,C2	391386	368753	347346	380062	338684	2648	378268	376019	1257
r16,F01,C8	187176	180132	178560	183475	174549	520	181756	181216	305
r16,F05,C8	423320	387580	376726	393541	369985	1236	393000	392189	741
r16,F10,C8	649121	599513	572014	610267	563050	1371	610267	610267	0
r17,F01,C1	374016	364784	364750	368841	361737	2014	368280	367439	768
r17,F05,C1	718135	693562	637970	717089	635288	4181	709084	707822	1729
r17,F10,C1	1041450	1006780	875442	1045940	872746	5351	1045940	1045940	0
r17,F01,C2	393608	382593	382555	388625	379451	1589	386383	385807	322
r17,F05,C2	786198	739859	706943	744146	702961	3179	740928	740928	0
r17,F10,C2	1162290	1138826	1002412	1126380	999737	3509	1110792	1105240	3104
r17,F01,C8	539817	530029	520755	535474	518847	792	535474	535474	0
r17,F05,C8	1348750	1229810	1188594	1241990	1185210	1381	1230106	1229770	307
r17,F10,C8	2227780	2024019	1902855	2094330	1895210	1544	2038944	2036760	2089
r18,F01,C1	864425	846152	831956	848636	828063	2922	845748	845748	0
r18,F05,C1	1640200	1689474	1526694	1615730	1523250	5930	1615730	1615730	0
r18,F10,C1	2399230	2484100	2123023	2280820	2122420	6570	2280820	2280820	0
r18,F01,C2	962402	942674	924095	947608	923085	2410	947447	947131	227
r18,F05,C2	1958160	2178442	1809148	1909340	1812830	3939	1909340	1909340	0
r18,F10,C2	2986000	3123686	2676729	2810300	2678790	3955	2810300	2810300	0
r18,F01,C8	1617320	1593899	1501925	1545030	1504650	1153	1537320	1537320	0
r18,F05,C8	4268580	4243216	3792924	3961280	3829060	1377	3961280	3961280	0
r18,F10,C8	7440780	7238683	6043152	6618400	6243990	1329	6618400	6618400	0

Table 6: Results for the 54 instances of Set **R**

Tables 7 and 8 display the detailed comparative results for instances in sets **C** and **R**, respectively. Columns BS/TS, BS/CPLEX, and BS/FS correspond to the improvement gaps of the average solution, noted BS, with respect to the Tabu Search of Pedersen et al. (2009), CPLEX, and the initial solution provided by the cutting-plane-based procedure we proposed, respectively (computed relative to BS). The last two columns display the gaps between BS and the lower bounds computed by CPLEX and the cutting-plane procedure, respectively.

INSTANCE	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
C20,230,200,V,L	-4.57%	-0.09%	0.00%	4.18%	4.72%
C20,230,200,F,L	-6.36%	0.64%	0.00%	5.28%	5.43%
C20,230,200,V,T	-2.09%	0.17%	-1.82%	2.47%	2.84%
C20,230,200,F,T	-6.24%	-0.94%	-1.07%	4.49%	4.75%
C20,300,200,V,L	-4.53%	1.75%	-2.38%	4.27%	4.67%
C20,300,200,F,L	-6.32%	0.96%	-0.16%	4.72%	5.24%
C20,300,200,V,T	-2.65%	0.26%	0.00%	0.93%	1.36%
C20,300,200,F,T	-2.58%	0.93%	0.00%	4.47%	4.64%
C30,520,100,V,L	-1.29%	0.83%	-1.13%	2.12%	2.46%
C30,520,100,F,L	-2.88%	1.86%	-4.40%	7.93%	7.86%
C30,520,100,V,T	-2.38%	0.36%	0.00%	0.92%	1.37%
C30,520,100,F,T	-0.65%	1.50%	-1.75%	6.62%	6.51%
C30,520,400,V,L	-3.69%	-0.01%	0.00%	1.87%	1.89%
C30,520,400,F,L	-5.47%	na	0.00%	4.83%	2.72%
C30,520,400,V,T	-2.96%	-0.22%	0.00%	1.04%	1.04%
C30,520,400,F,T	-3.92%	na	0.00%	2.32%	2.28%
C30,700,100,V,L	-0.59%	0.91%	-0.26%	1.22%	1.94%
C30,700,100,F,L	-3.05%	0.92%	-0.43%	4.01%	4.13%
C30,700,100,V,T	-2.83%	0.27%	-0.01%	1.39%	1.65%
C30,700,100,F,T	-2.84%	0.46%	-1.95%	2.67%	2.76%
C30,700,400,V,L	-4.36%	-2.29%	0.00%	1.79%	1.78%
C30,700,400,F,L	-12.91%	na	0.00%	4.07%	4.02%
C30,700,400,V,T	-5.30%	0.06%	0.00%	2.16%	2.14%
C30,700,400,F,T	-6.71%	na	0.00%	2.49%	2.47%
Average	-4.13%	-0.56%	-0.87%	4.34%	4.73%

Table 7: Improvement with respect to other solutions, C instances

Instance	BS/TS	BS/CPLEX	BS/FS	BS/LB_CPLEX	BS/LB
r13,F01,C1	0.44%	0.77%	0.00%	0.77%	0.95%
r13,F05,C1	0.42%	1.73%	-5.53%	3.18%	7.84%
r13,F10,C1	-0.21%	4.51%	-3.54%	4.52%	11.13%
r13,F01,C2	-2.54%	0.00%	0.00%	0.00%	1.11%
r13,F05,C2	-2.87%	2.43%	-0.44%	2.44%	7.14%
r13,F10,C2	-6.22%	1.84%	-2.74%	6.55%	10.25%
r13,F01,C8	-1.80%	1.16%	-1.49%	1.17%	3.53%
r13,F05,C8	-2.93%	1.13%	-0.09%	3.23%	5.19%
r13,F10,C8	-9.92%	1.86%	-0.26%	3.33%	5.93%
r14,F01,C1	-1.89%	0.20%	0.00%	0.21%	1.22%
r14,F05,C1	-1.15%	1.39%	-0.88%	5.91%	7.21%
r14,F10,C1	1.09%	5.08%	-0.27%	13.53%	14.47%
r14,F01,C2	-2.35%	0.54%	0.00%	0.55%	1.26%
r14,F05,C2	-5.84%	0.67%	0.00%	4.35%	5.11%
r14,F10,C2	-7.10%	-0.64%	0.00%	6.14%	5.69%
r14,F01,C8	-2.01%	0.55%	0.00%	1.14%	2.00%
r14,F05,C8	-5.10%	0.74%	0.00%	3.86%	4.71%
r14,F10,C8	-6.75%	0.74%	-0.61%	6.11%	6.32%
r15,F01,C1	-1.82%	0.31%	0.00%	0.46%	1.26%
r15,F05,C1	-7.24%	-0.02%	-0.22%	3.79%	3.97%
r15,F10,C1	-6.33%	-0.83%	0.00%	5.92%	6.15%
r15,F01,C2	-2.38%	0.15%	-0.39%	0.91%	1.25%
r15,F05,C2	-4.52%	-3.85%	0.00%	4.43%	4.08%
r15,F10,C2	-7.93%	-0.11%	-0.12%	7.39%	6.82%
r15,F01,C8	-2.87%	0.12%	-0.01%	0.52%	1.03%
r15,F05,C8	-4.66%	0.03%	0.00%	0.04%	0.52%
r15,F10,C8	-12.59%	0.12%	0.00%	0.13%	0.36%
r16,F01,C1	-1.13%	0.72%	-0.05%	0.72%	0.90%
r16,F05,C1	-0.28%	3.64%	-6.38%	6.42%	9.18%
r16,F10,C1	-7.25%	0.19%	0.00%	6.73%	8.42%
r16,F01,C2	-0.99%	1.01%	-10.66%	1.02%	1.57%
r16,F05,C2	-0.93%	5.50%	-1.39%	7.06%	10.33%
r16,F10,C2	-3.47%	2.52%	-0.47%	8.17%	10.46%
r16,F01,C8	-2.98%	0.89%	-0.95%	1.76%	3.97%
r16,F05,C8	-7.71%	1.38%	-0.14%	4.14%	5.86%
r16,F10,C8	-6.37%	1.76%	0.00%	6.27%	7.74%
r17,F01,C1	-1.56%	0.95%	-0.15%	0.96%	1.78%
r17,F05,C1	-1.28%	2.19%	-1.13%	10.03%	10.41%
r17,F10,C1	0.43%	3.74%	0.00%	16.30%	16.56%
r17,F01,C2	-1.87%	0.98%	-0.58%	0.99%	1.79%
r17,F05,C2	-6.11%	0.14%	-0.43%	4.59%	5.12%
r17,F10,C2	-4.64%	-2.52%	-1.40%	9.76%	10.00%
r17,F01,C8	-0.81%	1.02%	0.00%	2.75%	3.11%
r17,F05,C8	-9.65%	0.02%	-0.97%	3.37%	3.65%
r17,F10,C8	-9.26%	0.73%	-2.72%	6.67%	7.05%
r18,F01,C1	-2.21%	-0.05%	-0.34%	1.63%	2.09%
r18,F05,C1	-1.51%	-4.56%	0.00%	5.51%	5.72%
r18,F10,C1	-5.19%	-8.91%	0.00%	6.92%	6.94%
r18,F01,C2	-1.58%	0.50%	-0.02%	2.46%	2.57%
r18,F05,C2	-2.56%	-14.09%	0.00%	5.25%	5.05%
r18,F10,C2	-6.25%	-11.15%	0.00%	4.75%	4.68%
r18,F01,C8	-5.20%	-3.68%	-0.50%	2.30%	2.13%
r18,F05,C8	-7.76%	-7.12%	0.00%	4.25%	3.34%
r18,F10,C8	-12.43%	-9.37%	0.00%	8.69%	5.66%
Average	-4.03%	-0.24%	-0.83%	4.26%	5.23%

Table 8: Improvement with respect to other solutions, R instances