CIRRELT
Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport
Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

A Three-Stage Matheuristic for the Capacitated Multi-Commodity Fixed-Cost Network Design with Design-Balance Constraints

Vu Duc Minh
Teodor Gabriel Crainic
Michel Toulouse

May 2012

CIRRELT-2012-21

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

A Three-Stage Matheuristic for the Capacitated Multi-Commodity Fixed-Cost Network Design with Design-Balance Constraints

Vu Duc Minh[1,2], Teodor Gabriel Crainic[1,3,*], Michel Toulouse[1,4]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

[3] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[4] Department of Computer Science, Oklahoma State University, 700 North Greenwood Avenue, North Hall 328, Tulsa, OK 74106-0700, USA

Abstract. This paper proposes a three-stage matheuristic solution strategy for the capacitated multi-commodity fixed-cost network design problem with design-balance constraints. The proposed matheuristic combines exact and neighbourhood-based methods. The proposed tabu-search and path-relinking meta-heuristics cooperate to generate as many feasible solutions as possible. The two meta-heuristics incorporate new neighbourhoods, and computationally-efficient exploration procedures. The feasible solutions generated by the two procedures are then used to identify an appropriate part of the solution space where an exact solver intensifies the search. Computational experiments on benchmark instances show that the proposed algorithm finds good solutions to large-scale problems in a reasonable amount of time.

Keywords. Network design, design-balance constraints, tabu search, path relinking, matheuristic.

Acknowledgements. Partial funding for this project has been provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), by Calcul Québec and by the Fonds de recherche du Québec - Nature et technologies (FRQNT).

---

* Corresponding author: TeodorGabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2012

1 Introduction

This paper proposes a solution strategy for the *capacitated multi-commodity fixed-cost network design problem with design-balance constraints* (*DBCMND*), which is found in several important application domains, notably in transportation and the design of the service network of consolidation-based carriers when asset (resource) management issues are jointly considered (Crainic and Kim, 2007). The design of efficient solution methods for the DBCMND poses many challenges. The DBCMND is NP-hard and includes the capacitated multi-commodity fixed-cost network design problem (*CMND*), which is also NP-hard (Balakrishnan et al., 1997) as a particular case. The difficulty in addressing the DBCMND, as compared to the more classical CMND, is actually compounded by the strong interplay between the flow circulation and the design-balance property of the design. Pedersen et al. (2009) actually states that even the search for feasible solutions of the DBCMND is "far from trivial."

We propose a three-stage matheuristic procedure combining exact and neighborhood-based methods to address large DBCMND instances. We introduce two new heuristic methods, based on tabu-search and path-relinking principles, respectively, which cooperate to generate as many feasible solutions as possible. The tabu-search algorithm introduces a variant of the cycle-based neighborhood (Ghamlouche et al., 2003) that exploits the CMND substructure of DBCMND, and a minimum-cost maximum-flow formulation to extract feasible solutions when the CMND ones do not respect the design-balance constraints. A different neighborhood structure is proposed for the restricted path-relinking procedure, which aims to identify a large number of good feasible solutions in a computationally efficient way. The feasible solutions generated by the two procedures are then used to build statistical information and, thus, identify an appropriate part of the solution space where an exact solver intensifies the search. Experiments show that the proposed matheuristic performs well on problem instances with a large network and many commodities.

The rest of the paper is structured as follows. Section 2 states the problem and recalls the model formulation. Section 3 presents a brief literature review. Section 4 details the proposed algorithm. We present and analyze the experimental results in Section 5, and provide concluding remarks in Section 6.

2 Problem Statement and Model Formulation

The DBCMND is the combinatorial optimization problem where given a potential network in terms of its arcs and a set of node-to-node, origin-to-destination (OD), demands, one must select a set of arcs to satisfy demand at minimum total cost, while enforcing

the so-called design-balance constraints where the number of design arcs entering and exiting each node must be equal. Arcs are characterized by capacities limiting the total flow of commodities (OD demands) that use them. A fixed cost is incurred when an arc is selected (i.e., has positive flow), and a transportation cost is also paid for each unit of commodity flow passing though the arc.

In terms of consolidation-based carrier planning, nodes correspond to terminals and arcs to transportation services that could be operated between two terminals. The design-balance requirements correspond to so-called asset-management concerns. Assets typically refer to the resources e.g., crews or vehicles, needed to operate the selected transportation services, while asset management usually refers to the assignment of assets to services and their eventual repositioning.

This problem can be modeled by an oriented graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where the nodes, $\mathcal{N}$, represent the terminals, and the arcs, $\mathcal{A}$, represent the transportation services between these terminals. The set of commodities is noted $\mathcal{P}$ and represents the different products that must be transported between particular origin and destination nodes. For each commodity $p \in \mathcal{P}$, w_p stands for the amount of commodity p that must be moved from origin $o(p)$ to destination $d(p)$. A cost c_{ij}^p per unit of commodity p is associated with each arc (i, j) and represents the cost incurred when moving commodity p using this arc. The fixed cost of selecting and using arc $(i, j) \in \mathcal{A}$ is noted f_{ij}, and the capacity of the corresponding arc is noted u_{ij}.

Two sets of decision variables are associated with the formulation. A decision variable y_{ij} is associated with each arc: $y_{ij} = 1$ if arc (i, j) is used and $y_{ij} = 0$ otherwise. The continuous variables x_{ij}^p represent the flow distribution in the network in terms of the quantity of commodity p moved on arc (i, j). The arc-based mixed-integer formulation of the DBCMND Pedersen et al. (2009) is:

$$\min z(x, y) = \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \mathcal{A}} c_{ij}^p x_{ij}^p + \sum_{(i,j) \in \mathcal{A}} f_{ij} y_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{N}^+(i)} y_{ji} - \sum_{j \in \mathcal{N}^-(i)} y_{ij} = 0, \forall i \in \mathcal{N}, \tag{2}$$

$$\sum_{j \in N_i^+} x_{ij}^p - \sum_{j \in N_i^-} x_{ji}^p = d_i^p, \forall i \in \mathcal{N}, \forall p \in \mathcal{P}, \tag{3}$$

$$\sum_{p \in \mathcal{P}} x_{ij}^p \leq u_{ij} y_{ij}, \forall (i, j) \in \mathcal{A}, \tag{4}$$

$$x_{ij}^p \geq 0, \forall (i, j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{5}$$

$$y_{ij} \in \{0, 1\}, \forall (i, j) \in \mathcal{A}. \tag{6}$$

In this model, $\mathcal{N}_i^+ = \{j \in \mathcal{N} | (i,j) \in \mathcal{A}\}$ and $\mathcal{N}_i^- = \{j \in \mathcal{N} | (j,i) \in \mathcal{A}\}$ define respectively the outward and inward neighbors of node $i \in \mathcal{N}$, while d_i^p stands for the flow value of commodity p at node i, which equals w_p for $i = o(p)$, $-w_p$ when $i = d(p)$, and 0 otherwise. The objective function (1) minimizes the total system cost, i.e., the fixed cost of the selected arcs plus the routing cost of the commodities. Equations (2) are the design-balance constraints, which ensure that the total number of open arcs terminating at any node must equal the number of open arcs going out of that node. Constraints (3) ensure flow conservation for each node and each commodity. Constraints (4), often referred to as bundle or forcing constraints, state that the total flow on an arc (i,j) cannot exceed its capacity u_{ij} when selected and must be 0 if it is not selected. Constraints (5) and (6) are non negativity and integrality constraints for the decision variables.

We use the following notation borrowed from Pedersen et al. (2009) in the subsequent sections of this paper. We define the *node imbalance* $\psi^i = \sum_{j \in \mathcal{N}^+(i)} y_{ij} - \sum_{j \in \mathcal{N}^-(i)} y_{ji}$ to be the difference between the number of open outgoing arcs and the number of open incoming arcs at node i. The *total system imbalance* $\psi^{\mathcal{N}} = \sum_{i \in \mathcal{N}} |\psi^i|$ represents the total absolute value of all the node imbalances. The *maximum absolute imbalance* $\psi^{max} = max\,(|\psi^i|)$ is the largest absolute value among all the node imbalances and indicates the difficulty of achieving feasibility.

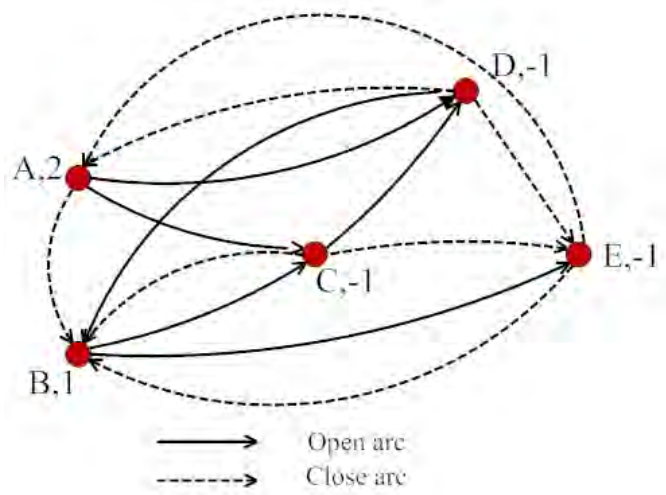

Figure 1: Design with node imbalance at each node.

Figure 1 illustrates these concepts on a five-node DBCMND instance. In this figure, the solid arcs represent the current design. We can see that node A has two outgoing design arcs and no incoming design arc, so $\psi^A = 2$; node D has two incoming design arcs and one outgoing design arc, so $\psi^D = -1$; etc. The total system imbalance is $\psi^{\mathcal{N}} = |\psi^A| + |\psi^B| + |\psi^C| + |\psi^D| + |\psi^E| = 6$. We can interpret the node imbalance at a given node i as the need for an additional $|\psi^i|$ empty vehicles moving in or out of that node. The cost of adding an empty vehicle is approximated by the average of the fixed

cost of the arcs in the network $\overline{f}$ or by the average of the fixed cost of the closed arcs in the network $\bar{f}_{closed}$.

3 Literature Review

Network design formulations have a wide range of applications in transportation, logistics, telecommunication, and production systems, in particular. Magnanti and Wong (1984), Minoux (1989), and Crainic (2000) present generic models and applications of network design together with solution methods. For specific application areas, see the reviews of Christiansen et al. (2007), Cordeau et al. (1998), and Crainic and Kim (2007) for optimization models in maritime, rail, and intermodal transportation, respectively.

Early work addressing asset-management issues in service network design includes Kim et al. (1999), Armacost et al. (2002), and Barnhart et al. (2002) for multimodal express network design, and Smilowitz et al. (2003) for multimodal package delivery with design-balance constraints for ground vehicles. In all these works, there must be an equal number of assets entering and leaving each node in the network. The solution methods use ad-hoc techniques that take advantage of the specific structure of the problems.

Pedersen et al. (2009) introduced the *design-balance* notation and the DBCMND. They proposed the first solution method for the DBCMND, a two-phase tabu-search algorithm including a local search algorithm to handle the design-balance constraints and restore feasibility. In the first phase of the tabu-search algorithm, neighbors are obtained by either adding or dropping arcs from the current solution, while satisfying the flow constraints but ignoring the design-balance constraints. A second phase seeks to convert the last solution of the first phase into a design-balanced feasible solution by generating solutions with a path-based neighborhood structure. At each iteration, the procedure obtains neighbors by adding or removing paths from the current solution while ignoring the flow constraints. The add/drop moves in this phase reduce the total system imbalance by two after each iteration, but many iterations may be required to obtain a design-balanced feasible solution.

Andersen et al. (2009a) and Andersen et al. (2009b) considered additional aspects of asset management such as the management and coordination of multiple fleets and fixed-length, cyclic schedules. This problem is denoted SNDAM. The authors compared the cycle-based and path-based formulations of SNDAM, concluding that cycle-based formulations contribute to efficient model solving. However, this study was based on an a priori enumeration of cycles and paths, and it cannot be directly applied to instances of realistic dimensions. Inspired by their previous work, Andersen et al. (2011) proposed the first branch-and-price approach using a cycle-based formulation for the SNDAM.

Chouman and Crainic (2010) proposed a hybrid of cutting planes and tabu-search procedures. The latter included a cycle-based neighborhood structure that directly addresses the design-balance requirements, but does not account for the corresponding flow-distribution feasibility. The need to verify this condition for each neighbor requires to solve the associated minimum-cost multi-commodity network flow problem, which is computationally very heavy.

Research into network design with asset management has thus been limited so far. On the one hand, the existing exact solution methods have limitations on the size of instances they may efficiently address. On the other hand, research on efficient meta-heuristics able to address larger instances has been extremely limited. Our goal is to contribute filling this gap, by returning to the generic DBCMND problem setting.

4 Proposed Matheuristic for the DBCMND

Our approach to the DBCMND exploits the natural complementarity between exact and neighborhood-based search methods. Exact solvers can only partially explore the solution space of large instances. To ensure success, the exploration should focus on promising regions of the solution space. We use heuristic methods to find feasible solutions from which we collect information that can be used to restrict the dimension of the problem. Then, the restricted problem is addressed using an exact solver.

Algorithm 1 displays the implementation sequence of this strategy. In the first phase, feasible solutions are identified using a tabu-search procedure initialized with a solution that satisfies all the flow constraints but may not satisfy the design-balance constraints. The feasible solutions found form the reference set of the path-relinking procedure that is used in the second phase to generate more feasible solutions. The solutions found in both phases are then analyzed in a third phase to determine which arcs can be fixed open or closed. Fixing the status of some variables reduces the size of the problem; an exact solver is then used to solve the reduced problem. The overall best solution of the algorithm is usually found in the third phase.

4.1 Tabu-search phase

The tabu-search meta-heuristic (Glover, 1989, 1990) that we propose aims to avoid the limitations in the contributions by Chouman and Crainic (2010) and Pedersen et al. (2009). Relative to the former, we relax the requirement that neighbours must satisfy the design-balance constraints when satisfying the flow constraints. In contrast to the latter, we seek solutions that are DBCMND feasible at each tabu-search iteration. Thus,

Algorithm 1 Proposed solution method template

Initialization. Solve a relaxed formulation of DBCMND to obtain an initial (possibly, design-balance unfeasible) solution.
Phase 1 - Tabu search. First exploration of the solution space to identify feasible solutions.
Phase 2 - Path Relinking. Using the solutions found by the tabu-search method as the reference set, generate new solutions to enrich the feasible solution set.
Phase 3 - MIP Intensification. Use statistical information obtained from the feasible solution set to fix variables and use a mixed-integer solver to address the corresponding reduced-size problem.
Output the best solution found.

once a solution in the neighborhood of the current solution has been selected as the new solution, our tabu-search method tries to restore feasibility with respect to the design-balance constraints by solving a minimum-cost maximum-flow problem. As supported by numerical experiments, compared to the search-based feasibility restoration approach of Pedersen et al. (2009), our approach is much faster and always identifies the set of arcs with the smallest possible cost.

4.1.1 Tabu-search neighbourhood exploration

The neighborhood definition is based on the cycle-based neighborhood of Ghamlouche et al. (2003) proposed to explore the space of the design variables of the CMND. This procedure identifies neighbors by first selecting a pair of nodes and two paths connecting those nodes, thus forming a cycle. Next, the flow on one path of the cycle is redirected to the other path (the alternative path) until the flow ceases on at least one arc.

Our cycle-based neighborhood procedure performs flow redirections that satisfy the flow distribution but do not guarantee that the design-balance constraints will be satisfied. Flow redirection is performed independently for each commodity that has flow on a given arc (i, j). This increases the chances of identifying neighbors that satisfy the flow distribution constraints because the entire demand can be satisfied via several paths.

We further enhanced the procedure of Ghamlouche et al. (2003) by considering other nodes, not always i and j, as the source and destination nodes of the alternative path, to facilitate the search for an alternative path for each commodity. Our flow redirection procedure is a loop that iterates on the set of commodities, identifying alternative nodes to nodes i and j for each commodity p that has flow on arc (i, j), and computing an alternative path to take the flow of commodity p from arc (i, j). We now describe the implementation of this cycle-based neighborhood exploration procedure.

Finding alternative source and destination nodes Let $\mathcal{P}_{ij} = \{p \in \mathcal{P} | x^p_{ij} > 0\}$ be the set of commodities with flow on arc (i, j). To close arc (i, j), we have to redirect this flow to other arcs while maintaining the flow constraints. We seek an alternative source node s^p for node i and an alternative destination node t^p for node j. For example, suppose we want to remove arc $(3, 4)$ with a flow of 2 units in the single-commodity graph of Figure 2. The flow on arc (3, 4) can be redirected to the subpath {(3, 7), (7,4)}. However, if the alternative source and destination nodes of nodes 3 and 4 are 2 and 5, then we can redirect the flow on arc $(3, 4)$ to subpath {(2, 3), (3, 7), (7, 4), (4, 5)}, or subpath {(2, 7), (7, 4)}, or subpath {(2, 3), (3, 7), (7, 5)}, etc. We cannot start from node 1 because arc $(1, 2)$ routes only 1 unit whereas arc $(3, 4)$ currently routes 2 units.

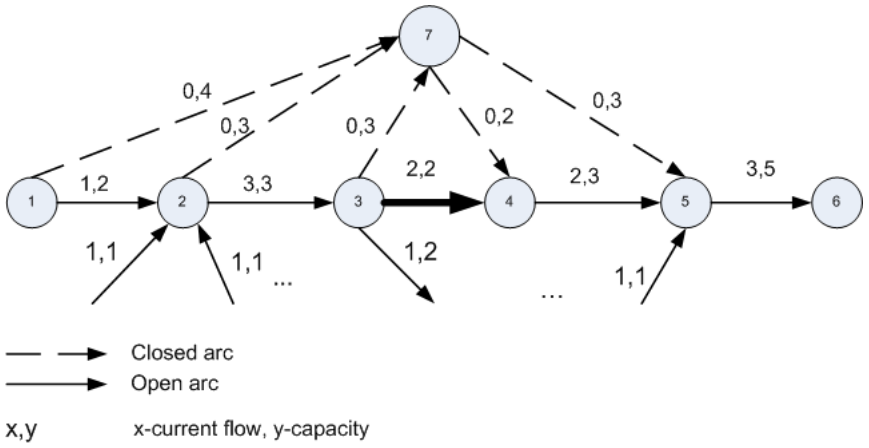

Figure 2: Illustration of alternative origin & destination cycle nodes

For each commodity p, we need to find nodes s^p and t^p such that the path connecting s^p and t^p passing through arc (i, j) can route at least x^p_{ij} units of commodity p. We find s^p (respectively t^p) using a breadth-first search algorithm starting from node i (j). We choose the node s^p (t^p) that is the furthest from node i (j), where the distance is measured by the number of arcs on each path. Intuitively, the greater the distance between s^p and i (j and t^p), the greater the probability of obtaining a feasible alternative path between s^p and t^p.

The procedure for identifying an alternative source node s^p is described in Algorithm 2. We initialize the queue with node i. When a node v is popped from the queue, the procedure tries to add another node u for which there is an arc (u, v) in the current design and the flow $x^p_{uv} \geq x^p_{ij}$. The last node v with no incoming arcs satisfying this condition is chosen as the source node s^p because we want to choose the node with the greatest distance from node i. The version of Algorithm 2 that finds an alternative destination node t^p is similar and is not shown here.

Residual graph definition Once the alternative source node s^p and destination node t^p are known, we proceed to compute a path connecting s^p and t^p to redirect the

Algorithm 2 Source_Search((i,j),p) Search for alternative source node s^p for arc (i,j)

1. $queue \leftarrow i$
2. **while** $queue \neq \varnothing$ **do**
3. $\quad v = queue.pop()$
4. $\quad$ **for each** $(u,v) \in \mathcal{A} \wedge y_{uv} = 1$ **do**
5. $\quad\quad$ **if** $x^p_{uv} \geq x^p_{ij}$**then**
6 $\quad\quad\quad queue.push(u)$
7. $\quad\quad\quad$ Save trace-back information
8. $\quad\quad$ **end if**
9. $\quad$ **end do**
10. $\quad$ **if** $queue = \varnothing$ **then** $s^p = v$, **break**
11. **end while**
12. **return** s^p

flow of commodity p from arc (i,j). A residual graph $\mathcal{G}^{\mathcal{R}}_p = (\mathcal{N}, \mathcal{A}^{\mathcal{R}}_p)$ is generated for this purpose for each commodity p. The set of arcs $\mathcal{A}^{\mathcal{R}}_p = \mathcal{A}^{\mathcal{R},open}_p \cup \mathcal{A}^{\mathcal{R},closed}_p$ is defined as follows. All closed arcs are included in the residual graph if their capacity u_{ij} is greater than or equal to the flow x^p_{ij} of commodity p on arc (i,j), i.e., $\mathcal{A}^{\mathcal{R},closed}_p = \left\{(k,l) \in \mathcal{A}\right\} | y_{kl} = 0 \wedge u_{kl} \geq x^p_{ij}\big\}$. Each open arc (k,l) (except arc (i,j)) is included in the residual graph if its residual capacity r_{kl} is greater than or equal to x^p_{ij} or if arc (k,l) is in $path_{s,t}$:

$$\mathcal{A}^{\mathcal{R},open}_p = \left\{(k,l) \in \mathcal{A} | \left(y_{kl} = 1 \wedge r_{kl} \geq x^p_{ij} \wedge (k,l) \notin path_{s,t}\right) \vee \left(y_{kl} = 1 \wedge (k,l) \in path_{s,t}\right)\right\}.$$

The residual capacity of (k,l) is $r_{kl} = u_{kl} + x^p_{ij} - \sum_{p \in \mathcal{P}} x^p_{kl}$, when $(k,l) \in path_{s,t}$, or $r_{kl} = u_{kl} - \sum_{p \in \mathcal{P}} x^p_{kl}$, otherwise.

The cost of a closed arc $(k,l) \in \mathcal{A}^{\mathcal{R},closed}_p$ is the fixed cost f_{kl} of opening the arc plus the cost of routing the commodity. The cost of an open arc $(k,l) \in \mathcal{A}^{\mathcal{R},open}_p$ is simply $c^p_{kl} x^p_{ij}$, the cost of routing the commodity. We use a classic shortest-path algorithm to find an alternative path between source node s^p and destination node t^p in the residual graph; the details are not given here.

Generating neighbors When the flow has been redirected successfully for all the commodities using arc (i,j), the cycle-based neighborhood procedure closes arc (i,j) and any other arcs with no residual flow. This yields a neighboring solution of the current solution x that we indicate by x_{ij}. The steps of the cycle-based neighborhood procedure are summarized in Algorithm 3. The current solution x is represented by its flow distribution $\mathcal{X}$ and its arc set $\mathcal{Y}$. The cost of x is indicated by $c(x)$, and $\mathcal{P}_{ij}$ stands the set of commodities routed over arc (i,j). The procedure first computes the cost of the neighboring solution x_{ij} after arc (i,j) has been removed from the current solution x (lines 1 and 2). Next, for each commodity $p \in \mathcal{P}_{ij}$ routed by arc (i,j), the appropriate source node s^p and destination node t^p are identified, the corresponding residual graph

is generated, and a shortest path is obtained to reroute the flow of commodity p (lines 4 to 7). If a feasible path is found, the cost, the flow distribution, and the design of the neighboring solution x_{ij} are updated (lines 9 and 10). The generation of the neighbor continues until all the commodities routed by arc (i, j) are rerouted. If it is not possible to reroute all the commodities, it is assumed that arc (i, j) cannot be removed and the neighbor for this arc cannot be generated.

Algorithm 3 Cycle-based neighborhood procedure Close_Open_Arc($\mathcal{X}, \mathcal{Y}, c(x)$, (i, j), $\mathcal{P}_{ij}$)

1. $c(x_{ij}) = c(x) - f_{ij} - \sum_{p \in \mathcal{P}_{ij}} c_{ij}^p x_{ij}^p$
2. $\mathcal{Y}_{ij} = \mathcal{Y} \backslash (i, j)$
3. **while** $\mathcal{P}_{ij} \neq \emptyset$ **do**
4. Choose a commodity $p \in \mathcal{P}_{ij}$
5. Find source node s^p and destination node t^p
6. Construct residual graph $\mathcal{G}^p$ for commodity p
7. Find shortest path π^p connecting s^p and t^p on $\mathcal{G}^p$
8. **if** $\pi^p \neq \emptyset$ **then**
9. Update $c(x_{ij})$ based on π^p
10. Update flow information of $\mathcal{X}_{ij}$ and design of $\mathcal{Y}_{ij}$
11. Remove commodity p, $\mathcal{P}_{ij} = \mathcal{P}_{ij} \backslash p$
12. **else return** *unfeasible*
13. **end while**
14. Close all arcs with no residual flow and update fixed cost
15. **return** x_{ij}.

4.1.2 Unfeasibility-monitoring scheme

To generate the entire neighborhood of the current solution, Algorithm 3 is called iteratively for each arc (i, j) of the current solution. The solutions returned are feasible with respect to the commodity flows but may not be feasible in terms of the design-balance constraints. Satisfying the design-balance constraints for all the solutions in the neighborhood would be too costly. Instead, we would like to choose the neighbor that has the best chance of becoming completely or almost completely feasible. We use the unfeasibility-monitoring scheme proposed by Pedersen et al. (2009) for this purpose. A penalty is used to estimate how far the solution is from feasibility with respect to the design-balance constraints.

Using the vocabulary of transportation applications, the node imbalance at a given node i may be interpreted as the need to add $|\psi^i|$ empty-vehicle services in or out of that node. The cost $\widetilde{f}$ of adding an empty vehicle is approximated as the product of the average $\overline{f}$ of the fixed cost of the arcs of neighboring solution x_{ij} and an empirical scaling parameter τ used to control the importance of the penalty when evaluating neighboring

solutions. Then, to obtain a penalty that increases non-linearly with the unfeasibility of the solution, we multiply the estimated cost of an "empty-vehicle service" (arc), the total system imbalance of solution x_{ij}, and the maximum node imbalance of x_{ij}, i.e., the penalty $P_{ij} = \widetilde{f}\psi^{\mathcal{N}}\psi^{max}$, where $\widetilde{f} = \tau\overline{f}$. This penalty is added to the cost of neighboring solution x_{ij} yielding its *total system cost* $V_{ij} = c(x_{ij}) + P_{ij}$. When comparing neighboring solutions, P_{ij} provides the means to skew the evaluation towards feasibility, a neighbor with a relatively high total system cost that is close to feasibility may be preferred to one with a lower total system cost that is further from feasibility.

4.1.3 Design-balance feasibility

The neighbor that minimizes the total system cost is thus selected and then, if required, one attempts to restore feasibility with respect to the design-balance constraints.

Consider a current solution x that violates the design-balance constraints (Figure 1 illustrates such a case). We seek to reduce the total system imbalance of x to zero by introducing paths that contain only arcs that are closed in the current solution. These paths connect nodes with opposite imbalance signs. To reduce the total system imbalance, these paths will start from nodes with negative imbalances and end in nodes with positive imbalances. We add the paths that give the greatest reduction in the total system imbalance. When several paths satisfy this criterion, we choose the one with the smallest cost. The arcs on these paths are found by solving a minimum-cost maximum-flow problem as described below.

Let $\mathcal{N}^+ = \{u \in \mathcal{N} | \psi^u > 0\}$ and $\mathcal{N}^- = \{u \in \mathcal{N} | \psi^u < 0\}$ be the subsets of nodes in x that have positive and negative imbalances, respectively. Recall that in a network, the total positive and negative imbalances are the same, i.e., $\left|\sum_{u\in\mathcal{N}^+} \psi^u\right| = \left|\sum_{u\in\mathcal{N}^-} \psi^u\right|$, implying $\sum_{u\in\mathcal{N}^+} \psi^u + \sum_{u\in\mathcal{N}^-} \psi^u = 0$. Let $\mathcal{G}^{\mathcal{F}} = (\mathcal{N} \cup \{s,t\}, \mathcal{A}^{\mathcal{F}} \cup \mathcal{A}^{\mathcal{G}})$ be the *minimum-cost maximum-flow graph* corresponding to solution x. $\mathcal{A}^{\mathcal{F}} = \{(i,j) \in \mathcal{A} | y_{ij} = 0\}$ is the set of closed arcs in solution x. The cost of each arc in $\mathcal{A}^{\mathcal{F}}$ is set to the fix cost of the corresponding arc in the original problem, while its capacity is fixed to 1. Nodes s and t are artificial nodes connected to nodes in $\mathcal{N}$ by arcs in $\mathcal{A}^{\mathcal{G}}$ as follows: There is an arc $(s,u) \in \mathcal{A}^{\mathcal{G}}$ with cost 0 and capacity $-(\psi^u)$ for every $u \in \mathcal{N}^-$. Similarly, there is an arc $(u,t) \in \mathcal{A}^{\mathcal{G}}$ with cost 0 and capacity ψ^u for every $u \in \mathcal{N}^+$. The graph of the problem displays now the classic structure of a minimum-cost maximum-flow problem.

Figures 1 and 3 illustrate this procedure. The latter displays the minimum-cost maximum-flow graph corresponding to the unfeasible solution in the former. Arcs (s,D), (s,C), and (s,E) in Figure 3 have capacity 1 and connect the artificial node s to nodes in the solution of Figure 1 with negative imbalances. Arcs (A,t) and (B,t), with respective capacities of 2 and 1, connect nodes A and B with positive imbalances in the solution of Figure 1 to the artificial node t. All the other arcs in Figure 3 form the set $\mathcal{A}^{\mathcal{F}}$ of

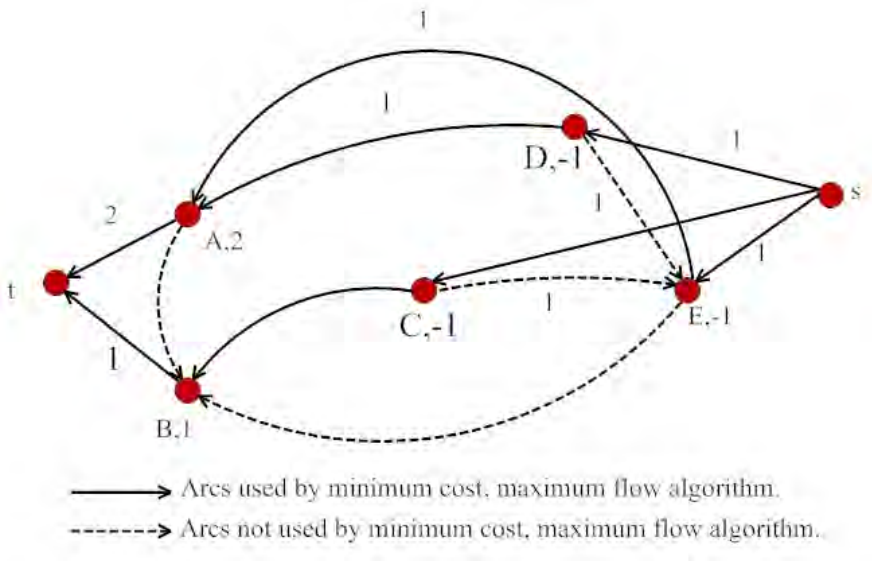

Figure 3: The equivalent minimum-cost maximum-flow network

non-artificial arcs in $\mathcal{G}^{\mathcal{F}}$ and correspond to closed arcs in the solution of Figure 1.

The minimum-cost maximum-flow algorithm then seeks to route $\sum_{u \in \mathcal{N}^+} \psi^u$ units of flow between s and t in $\mathcal{G}^{\mathcal{F}}$. Discarding the arcs connecting the artificial nodes s and t, the solution of the minimum-cost maximum-flow algorithm yields a set of arcs $\mathcal{A}' \subseteq \mathcal{A}^{\mathcal{F}}$, illustrated in Figure 3 by the non-artificial solid arcs, which are added to the design of the current solution x to satisfy the design-balance constraints (illustrated in Figure 4). Notice that, since the capacity of the arcs in $\mathcal{A}^{\mathcal{F}}$ is set to 1, the flow routed over $\mathcal{G}^{\mathcal{F}}$ decomposes into several disjoint paths. Each path starts from a node u with $\psi^u < 0$ and ends in a node v with $\psi^v > 0$. Moreover, for each node $z \in \mathcal{N}$ where $\psi^z = 0$, the numbers of new incoming and outgoing arcs are equal because of the flow conservation constraints. Therefore, the addition of the new arcs of $\mathcal{A}'$ does not change the balance at node z when $\psi^z = 0$ in the current solution.

The minimum-cost maximum-flow algorithm allows us to find the maximum flow that can be routed from s to t over $\mathcal{G}^{\mathcal{F}}$ at the smallest cost, which is equal to the cost of the arcs in $\mathcal{A}^{\mathcal{F}}$ used to route the flow. The maximum flow is the number of disjoint paths that we can obtain in $\mathcal{A}^{\mathcal{F}}$ by connecting a node u and a node v where $\psi^u < 0$ and $\psi^v > 0$. Each path reduces the total system imbalance by two, and the maximum flow corresponds to the smallest total system imbalance that we can obtain by adding arcs to the current design. This allows us to directly obtain a good feasible solution from an unbalanced and unfeasible neighboring solution, if such a feasible solution exists for the unbalanced solution. Even if we do not obtain a feasible solution after matching the nodes with opposite signs, the added arcs reduce the total system imbalance as much as possible, bringing the solution closer to feasibility with respect to the design-balance

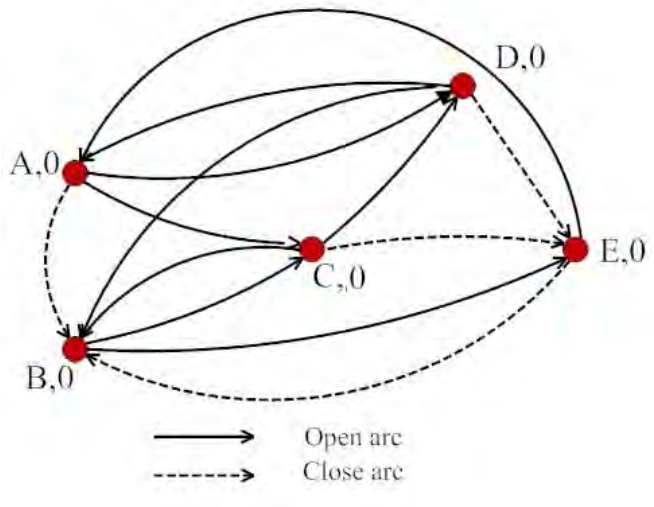

Figure 4: Feasible solution with arcs used by minimum-cost maximum-flow algorithm

constraints.

4.1.4 The Tabu Search algorithm

The evolution of the tabu-search algorithm is controlled through a short-term tabu list recording each arc (i, j) that originated a move, i.e., flow has been diverted from arc (i, j) and the corresponding neighboring solution x_{ij} has been selected as the new solution. The tabu list prevents the reopening of arc (i, j) for a predefined number of tabu-search iterations, and it is used when the algorithm computes alternative paths, as well as by the minimum-cost maximum-flow algorithm. In the former case, a tabu arc cannot be used to divert the flow from another arc, even when this prevents finding an alternative path. With respect to the latter, closed arcs in the tabu list cannot be reopened to satisfy the design-balance constraints. In this case, the tabu list prevents the cycling that might otherwise occur.

Arcs that are opened by the minimum-cost maximum-flow algorithm to satisfy the design-balance constraints are also placed in the tabu list. The tabu status of these arcs prevents their rapid removal from the solution by tabu-search moves. This induces the tabu-search algorithm to explore regions of the solution space where the design-balance constraints are satisfied.

Algorithm 4 gives the main steps of the proposed tabu-search algorithm, which stops after executing a predefined number of iterations or when a maximum computational time is reached. The search space is the set of design solution vectors that are feasible with respect to all the flow constraints and variable constraints. Moves maintain the flow

and variable constraints, but new solutions may violate the design-balance constraints. However, only solutions that satisfy the design-balance constraints are stored in the solution set. The solution set becomes the input to the restricted path-relinking phase of our algorithm.

Algorithm 4 Tabu-search algorithm for the DBCMND

Initialization Solve a relaxation of DBCMND to obtain an initial (potentially unfeasible) solution;

Exploration & Feasibility. While *stopping conditions* not satisfied, **do**

1. For each arc of the current solution, generate a neighboring solution (Section 4.1.1);
2. Select the neighbor that minimizes the total system cost as the new solution;
3. Solve the minimum-cost maximum-flow problem (Section 4.1.3) to obtain a set of arcs that reduces the total system imbalance of the new current solution;
4. Add these arcs to the current solution;
5. Update the tabu list;
6. Solve the corresponding multi-commodity minimum-cost flow problem using the new design;
7. If a DBCMND feasible solution is obtained, store it in the solution set and update the global optimal solution, if necessary.

4.2 Path-relinking phase

Despite a neighborhood structure that identifies rather easily good candidate solutions at each tabu-search iteration, and despite the fact that the minimum-cost maximum-flow algorithm facilitates obtaining DBCMND feasible solutions, we have observed that the number of feasible solutions found by tabu-search is still too small to accurately determine regions of the solution space in which to intensify the search. The goal of the restricted path-relinking phase is to enrich this set starting from the feasible solutions found by tabu-search algorithm.

Path Relinking (Glover, 1997; Glover et al., 2000) is a population-and-neighborhood-based meta-heuristic that operates on a set of solutions, called the *reference set*, generating paths between solutions in this set that yield new solutions. Each iteration of this algorithm starts from an *initial solution* and builds a path toward a *guiding solution*, by performing moves that progressively introduce into the initial solution the desirable *attributes* of the guiding solution. This exploration allows the search to perform moves

that may be unattractive according to the objective function value but appear essential for reaching solutions with given characteristics.

The design of a path-relinking algorithm involves specifying the reference set, how the initial and guiding solutions are selected in the reference set, which guiding attributes need to be introduced into the initial solution, and at least a neighborhood defining how the path from the initial solution to the guiding solution is to be built.

Reference set and initial and guiding solutions. The reference set contains all the feasible solutions found during the tabu-search phase if the number is less than or equal to a parameter l_r, or the best l_r solutions, otherwise.

We have implemented a backward path-relinking procedure where the initial solution is the best solution in the reference set, and the guiding solution is the second-best solution. When a path has been completed between the two, the guiding solution is removed from the reference set. We have observed that backward path-relinking tends to find better solutions than a forward version. The former explores more thoroughly the neighborhood of the initial solution, the best solution, where one expects to find good solutions, while the increasingly restricted neighborhood along the path may not provide the same opportunity. In the following, the "current solution" refers to the initial solution at the current iteration of the algorithm, as transformed in all previous iterations.

Neighbourhood structure and guiding attributes The guiding attributes are arcs that are open or closed in the guiding solution. Therefore, we seek to introduce into the current solution arcs that are open in the guiding solution but closed in the initial solution, and we seek to remove arcs that are closed in the guiding solution but open in the initial solution.

We define a neighborhood and move that perform both actions. A neighbor therefore is a solution obtained from the current solution by opening at least one closed arc in the current solution that is open in the guiding solution and by closing at least one open arc that is closed in the guiding solution. This move is performed using a path-exchange neighborhood procedure: the path removed from the current solution contains only arcs that are open in the current solution and closed in the guiding solution, and the alternative path introduced contains only arcs that are closed in the current solution with at least one of them being open in the guiding solution.

Note that the path-relinking procedure has been designed to be fast while generating reasonably good feasible solutions, i.e., feasible solutions that are not necessarily better than those found by the tabu-search algorithm. For this reason, the exploration of the neighborhood is restricted to only one neighbor at each iteration, which becomes the new

current solution. The procedure stops when it fails to generate a neighbour, and restarts with a new guiding solution.

To perform the path exchange, we start by finding a path to remove. First, we identify an arc (i, j) such that (i, j) is open in the current solution but closed in the guiding solution. Next, similarly to the tabu-search neighborhood, we seek to identify two nodes s and t, the start and end nodes of the path to be removed. Starting from i, a simple backward search is performed to find a source node s such that the path between i and s contains only open arcs that do not belong to the guiding solution. The same procedure is applied to j to find a node t such that the path between j and t contains only arcs that are closed in the guiding solution. Note that we do not consider the flow-conservation constraints when seeking the nodes s and t. We remove the path that contains the arcs from s to i, j to t, and arc (i, j). If several nodes s and t have been identified, we choose among them randomly, instead of choosing the furthest node as in the tabu-search neighborhood.

An alternative path is a path that reconnects the end points s and t of a removed path. The alternative path is computed in a graph $\mathcal{G}^{PR} = \left(\mathcal{N}, \mathcal{A}^{PR}\right)$, defined on the set of closed arcs in the current solution $\mathcal{A}^{PR} = \{(i, j) \in \mathcal{A} | y_{ij} = 0\}$. The cost of each arc $(i, j) \in \mathcal{G}^{PR}$ is the product of its capacity u_{ij} and the maximum-flow cost on arc (i, j), $c_{ij}^{PR} = f_{ij} + max\{c_{ij}^{p}\}_{p \in \mathcal{P}} u_{ij}$. This is an upper bound on the contribution to the objective value of the current solution from the arc (i, j).

Define $\mathcal{G}^{PR,inverse} = \left(\mathcal{N}, \mathcal{A}^{PR,inverse}\right)$, the reverse graph obtained by reversing the direction of all arcs in $\mathcal{A}^{PR}$. Intuitively, the best way to find an alternative path is to solve a minimum-cost maximum-flow problem. However, this may be costly because we need to solve this problem for each arc $(u, v) \in \mathcal{G}^{PR}$ that is also an arc of the guiding solution. We use a different approach to reduce the computational time. Using the nodes s and t of the path to be removed, we compute the shortest paths from s to all the other nodes in the graph $\mathcal{G}^{PR}$. The results are stored in array $d^s[..]$. We also find the shortest paths from t to all the other nodes in the graph $\mathcal{G}^{PR,inverse}$, and the results are stored in array $d^t[..]$. Finally, we find an arc (u, v) that is closed in the current solution but open in the guiding solution and that minimizes the sum $d^s[u] + d^t[v] + f_{uv}$, where $d^s[u]$ is the cost of the shortest path from node s to node u and $d^t[v]$ is the cost of the shortest path from node v to node t. The path formed by the arc (u, v) and the corresponding shortest paths is added to the current solution. Note that we can use the all-pairs shortest-path algorithm and calculate all the necessary shortest paths just once for a given current solution. Algorithm 5 presents this neighborhood-exploration procedure.

A new current solution is then obtained by replacing path $s..i{-}j..t$ with path $s..u{-}v..t$. This solution is always feasible with respect to the design-balance constraints. Actually, the path that is removed leaves s and t unbalanced. However, when these two nodes are reconnected by an alternative path, they become balanced once again. All the other

Algorithm 5 Path_exchange_neighbourhood($\mathcal{G} = (\mathcal{N}, \mathcal{A}), i, j$)

Find s, t and identify $path^{s,t}$ connecting s and t passing through arc (i, j);
Generate graph $\mathcal{G}^{PR} = \left(\mathcal{N}, \mathcal{A}^{PR}\right)$;
Find the shortest paths from s to all nodes u in $\mathcal{G}^{PR}$, save these in $d^s[u]$ and trace back information;
Find the shortest paths from t to all nodes v in $\mathcal{G}^{PR,inverse}$, save these in $d^t[v]$ and trace back information;
Find a closed arc (u, v) of the current solution that is an open arc in the guiding solution and minimizes $d^s[u] + d^t[v] + f_{uv}$;
Return (u, v) and the two paths connecting s and t.

nodes on the removed and alternative paths see the addition or removal of exactly one entering arc and one exiting arc.

The restricted path-relinking procedure is described in Algorithm 6. The reference set is a set of best feasible solutions generated during the tabu-search phase, and the initial solution is the best solution in this set. The main loop iterates over the elements of the reference set, each being discarded after it has been used as the "guide" for an iteration of this loop. The inner loop then generates solutions on the path between the current pair of solutions, exploring the path-exchange neighborhood (Algorithm 5) between the paired solutions. The arcs to close (set $ArcsToClose$) are those in the current solution that do not appear in the guiding solution. Since a move opens at least one arc in the current solution that is closed in the guiding solution and closes at least one arc that is open in the guiding solution, the set $ArcsToClose$ controls the trajectory of the inner loop.

Notice that, while the design obtained from the path-exchange move is design-balance feasible, it may not satisfy the flow-balance constraints. The new solution is then computed by solving a restricted DBCMND problem in which all the arcs of the current solution are fixed. It is necessary to solve a DBCMND problem rather than simply a minimum-cost flow problem because the likelihood of obtaining an unfeasible flow with the current design is quite high. Solving the DBCMND problem adds new arcs to the current design until the flow becomes feasible. The resulting solution is added to the solution set, but it is not used to update the current solution. Note that the arcs added while solving the restricted DBCMND help to diversify the solution set built during the path-relinking phase.

4.3 Intensification phase

The goal of this last phase is to find better solutions by intensifying the search in promising regions of the solution space. The identification of such promising regions is obtained

Algorithm 6 Main path-relinking procedure

Initialization.

Fill the reference set $\mathcal{R}$ with the best feasible solutions yielded by the tabu-search procedure;

Set $InitialSolution$ to the best solution in $\mathcal{R}$;

while $\mathcal{R} \neq \emptyset$ **do**

Set $GuidingSolution$ to the second-best solution in $\mathcal{R}$, and remove it from the reference set;

Set $CurrentSolution = InitialSolution$;

Define $ArcsToClose$ to be the set of open arcs in the $CurrentSolution$ that are not open in the $GuidingSolution$;

while $ArcsToClose \neq \emptyset$ **do**

Randomly select an arc $(i, j) \in ArcsToClose$, and set $ArcsToClose = ArcsToClose \setminus (i, j)$;

Apply the Path_exchange_neighbourhood procedure for arc (i, j) (Algorithm 5);

if a feasible alternative path is found **then**

for All arcs (k, l) in the alternative path **do**

Add arc (k, l) to $CurrentSolution$;

Set up a restricted DBCMND problem by fixing all the design arcs in the current solution and solve the resulting MIP to identify a flow-feasible solution;

if A feasible solution is found **then**

Add it to the solution set;

end if

end for

end if

end while

end while

by fixing to open or closed certain arcs according to their frequency in good solutions identified during the search. Two restricted DBCMND problems are thus defined, which are then solved by an exact MIP solver.

Let S be the solution set generated by the tabu-search and restricted path-relinking procedures, $c(x)$ the cost of a solution $x \in S$, and $maxcost^S$ the cost of the worst solution in the set. Define the $rate_{ij}$ of an arc (i, j) as

$$rate_{ij} = \sum_{x \in S} y_{ij}^x * \left(1 - \frac{c(x)}{maxcost^S}\right), \forall (i,j) \in \mathcal{A} \tag{7}$$

The rate expression assigns greater weights to arcs that either belong to good solutions in the set or that appear in many solutions of the set. Rates are then normalized in the range $[0, 1]$ by dividing them by the maximum value $rate_{max} = max\{rate_{ij} | (i, j) \in \mathcal{A}\}$. An arc (i, j) is then fixed to open or closed based on its normalized $rate_{ij}$ value.

The first restricted problem is defined by assuming that arcs with a high rating are more likely to appear in the optimal solution. Consequently, the arcs for which $rate_{ij}$ is greater than a certain threshold α are fixed to open. Let x_{open} be the optimal solution of the corresponding restricted DBCMND problem. Symmetrically, we assume that arcs with a low rating are unlikely to be part of the optimal solution, and fix to closed the arcs with a $rate_{ij}$ lower than a certain threshold β. Obviously, we do not fix arcs appearing in x_{open}. The resulting restricted DBCMND is then solved to optimality. The final solution of the proposed matheuristic is then the best solution among those generated by the tabu-search, path-relinking, and intensification phases.

5 Computational Results

The algorithm we propose, and that we call *TS-PR* in the following, is validated empirically using the problem instances used in Pedersen et al. (2009), as well as in a number of other papers in the literature (e.g., Ghamlouche et al., 2003, 2004). There are two sets of instances identified as R and C, respectively. Both sets are general transshipment networks with no parallel arcs and one commodity per origin-destination pair. The instances vary in size (nodes, arcs, commodities). Furthermore, the fixed costs, variable costs, and capacities vary. The same arc unit cost is used for all the commodities.

The algorithm is coded in C++ using Microsoft Visual Studio 2008. CPLEX 12.1 was used to solve the flow problem and restricted MIPs. All the computing times reported in this section were obtained from computers with the AMD Dual-Core Opteron 64-bit microprocessor with 2.4 Ghz and 16 GB Ram, under the Linux operating system.

We first discuss the calibration of the algorithm parameters, followed by an analysis of the behaviour of each phase and, finally, by a comparative analysis of its performance in relation to results in the literature.

5.1 Parameter calibration

The calibration is based on the same ten instances used by Pedersen et al. (2009). These instances display various combinations of characteristics and are representative of the overall set of instances. They include C and R instances. For convenience, these instances are listed in Table 1. The "Capacity Ratio" refers to the ratio of the commodity demand to the total network capacity; a "Tight" capacity ratio indicates a limited capacity with respect to the total demand while a "Loose" capacity ratio indicates a surplus of capacity. The "Cost ratio" is the ratio of the fixed cost to the variable cost of the arcs.

Instance	Nodes	Arcs	Commodities	Capacity Ratio	Cost Ratio
R10,F05,C2	20	120	40	Medium	Medium
R12,F10,C2	20	120	200	Medium	High
R13,F01,C8	20	220	40	Tight	Low
R15,F10,C8	20	220	200	Tight	High
C20,230,200,F,L	20	230	200	Loose	High
R16,F10,C1	20	314	40	Loose	High
R17,F01,C1	20	318	100	Loose	Low
R18,F05,C2	20	315	200	Medium	Medium
C30,520,100,V,T	30	519	100	Tight	Low
C100,400,30,F,L	100	400	30	Loose	High

Table 1: Problem instances used for calibration

We have calibrated the five parameters shown in Table 2. Each parameter was tested with two values indicated in the "Values Tested" column. "Value selected" displays the selected value.

Parameter	Values Tested	Value Selected
Penalty scaling factor	0.2, 0.5	0.2
Length of tabu list	10, 15	10
Length of reference set	10, 20	20
Intensification open parameter	0.8, 0.9	0.8
Intensification closed parameter	0.1, 0.2	0.2

Table 2: Calibrated parameters

5.2 Internal performance analysis

Tables 3 and 4 summarize our results for the 78 problem instances. Each instance was solved ten times with the same parameter values to account for stochastic variations in the algorithm. Each table is divided into three sections labeled "TS", "PR", and "Intensification" for the three phases of the algorithm, the tabu search, path relinking, and intensification phases, respectively. The columns labeled "T" give the average computational time in minutes. The maximum running time for the tabu-search phase was 1 hour, for the path relinking was 2 hours, and 1 hour for each restricted DBCMND of the intensification phase. The total maximum running time for the complete algorithm is thus 5 hours. These numbers were chosen so that the results are easy to compare with those of other methods. The columns labeled "S" give the average number of feasible solutions found. We notice that the restricted path-relinking phase achieves its stated goals. It identified many new feasible solutions at a small computational cost, without loosing too much on solution quality.

In the two tables, the columns labelled "Best" give the best solutions obtained over the ten runs. Table 5 reports the average standard deviation over these repetitions. The values are very low, which underlines the robustness of the algorithm we propose.

The results in Table 6 focuses the analysis on the intensification phase by reporting the average difference between the optimal solutions found during intensification for the restricted formulation compared with the best solution found by the tabu-search in TS-PR. Columns "O. Int./TS" and "C. Int./TS" report these figures for the two cases, i.e., when the intensification is performed by fixing arcs to open and closed, respectively. The figures indicate that improved solutions are found during intensification, which emphasizes the importance of the procedure within the complete algorithm. We notice that the second case, fixing closed arcs, finds better solutions. This is because arcs that are open during the first intensification case are not fixed to closed during the second one even if they have a small $rate_{ij}$ value.

5.3 Comparative analysis

We compare the results of the meta-heuristic we propose and those from the state-of-the-art tabu-search algorithm of Pedersen et al. (2009) obtained after 1 hour of computation, as well as those of the MIP algorithm of CPLEX 12.1 obtained after 1 and 5 hours, and with the lower bound obtained after 10 hours. Table 7 provides a general view of the effectiveness of TS-PR by displaying the average improvement gap (negative values indicate better results) and number of improved solutions, respectively, obtained by TS-PR with respect to those of other methods. Columns "TS-PR/P-TS", "TS-PR/CPLEX 1 h" and "TS-PR/CPLEX 5 h" report these measures relative to the tabu-search algorithm of

Instance	TS			PR			Intensification	
	T	S	Best	T	S	Best	T	Best
C20,230,200,V,L	60	10	100,708	8	85	101,640	23	97,274
C20,230,200,F,L	60	6	145,958	5	40	148,051	27	139,395
C20,230,200,V,T	60	10	102,906	11	43	104,374	7	100,720
C20,230,200,F,T	60	7	142,038	6	33	151,571	18	138,962
C20,300,200,V,L	60	15	80,166	31	95	80,162	71	77,584
C20,300,200,F,L	60	9	124,846	5	47	127,816	45	119,987
C20,300,200,V,T	60	13	77,692	23	68	78,384	7	76,450
C20,300,200,F,T	60	11	119,384	19	88	123,211	62	111,776
C30,520,100,V,L	60	29	55,002	21	185	55,117	4	54,783
C30,520,100,F,L	60	21	102,735	41	187	103,644	17	100,098
C30,520,100,V,T	60	16	53,313	42	166	53,516	1	53,035
C30,520,100,F,T	60	13	102,484	50	109	105,938	55	101,412
C30,520,400,V,L	60	17	118,627	105	157	117,824	56	115,528
C30,520,400,F,L	60	22	155,270	67	110	203,505	19	153,409
C30,520,400,V,T	60	19	119,795	91	127	118,965	59	117,226
C30,520,400,F,T	60	26	156,694	15	123	214,384	61	155,906
C30,700,100,V,L	60	25	48,879	13	91	49,185	1	48,807
C30,700,100,F,L	60	23	61,745	50	146	62,838	43	61,408
C30,700,100,V,T	60	24	47,141	49	155	47,032	26	46,812
C30,700,100,F,T	60	29	56,810	71	225	57,058	21	56,237
C30,700,400,V,L	60	35	101,423	53	125	139,217	23	100,589
C30,700,400,F,T	60	21	141,037	5	185	142,046	15	141,037
C30,700,400,V,T	60	14	108,481	100	116	108,710	62	97,875
C30,700,400,F,T	60	13	134,320	3	60	134,707	67	133,686

Table 3: Computational results on C instances

Instance	TS			PR			Intensification	
	T	S	Best	T	S	Best	T	Best
R13,F01,C1	60	16	148,425	1	78	150,511	1	147,349
R13,F05,C1	60	19	289,535	2	113	293,431	1	277,891
R13,F10,C1	60	15	403,490	2	115	419,908	1	385,396
R13,F01,C2	60	18	157,743	1	80	158,392	1	155,887
R13,F05,C2	60	18	306,783	2	128	311,442	1	295,655
R13,F10,C2	60	13	456,269	1	93	467,241	2	436,773
R13,F01,C8	60	18	219,105	1	82	223,460	1	218,787
R13,F05,C8	60	11	500,403	2	98	509,431	9	492,959
R13,F10,C8	60	18	813,589	1	136	831,592	2	789,641
R14,F01,C1	60	23	437,836	8	140	440,335	1	424,039
R14,F05,C1	60	16	853,765	4	121	861,162	1	784,626
R14,F10,C1	60	6	1,214,782	2	28	1,242,096	11	1,131,900
R14,F01,C2	60	14	455,609	4	81	458,743	1	454,031
R14,F05,C2	60	13	914,839	7	109	931,059	28	883,051
R14,F10,C2	60	9	1,378,765	3	57	1,415,336	22	1,308,030
R14,F01,C8	60	16	714,841	7	71	717,294	40	703,259
R14,F05,C8	60	9	1,743,116	2	39	1,831,803	46	1,695,160
R14,F10,C8	60	10	2,874,717	15	126	2,940,505	94	2,757,660
R15,F01,C1	60	6	1,033,662	5	31	1,038,819	2	1,019,390
R15,F05,C1	60	9	2,156,433	7	60	2,199,282	47	2,017,150
R15,F10,C1	60	4	3,362,675	4	18	3,673,775	61	2,985,570
R15,F01,C2	60	12	1,177,198	15	66	1,180,909	12	1,174,520
R15,F05,C2	60	3	2,785,075	8	26	2,857,848	78	2,571,880
R15,F10,C2	60	5	4,430,712	8	44	4,528,368	102	4,017,230
R15,F01,C8	60	4	2,408,210	18	10	2,408,210	39	2,408,210
R15,F05,C8	60	7	5,874,704	24	29	5,916,736	65	5,796,510
R15,F10,C8	60	7	9,205,777	21	28	9,306,490	4	9,129,360
R16,F01,C1	60	14	141,352	1	54	143,649	1	140,082
R16,F05,C1	60	23	260,685	2	96	272,657	1	248,703
R16,F10,C1	60	11	371,854	2	81	395,754	1	345,243
R16,F01,C2	60	22	143,345	3	103	145,190	1	142,605
R16,F05,C2	60	14	264,345	2	84	282,548	1	261,937
R16,F10,C2	60	12	375,258	3	105	388,991	4	363,999
R16,F01,C8	60	31	181,350	5	146	186,039	1	180,132
R16,F05,C8	60	15	398,082	2	119	407,366	2	391,796
R16,F10,C8	60	18	629,107	2	142	643,342	11	604,430
R17,F01,C1	60	15	372,914	10	83	381,485	1	366,492
R17,F05,C1	60	13	695,550	13	121	737,784	60	676,528
R17,F10,C1	60	13	1,037,684	12	123	1,085,083	59	953,009
R17,F01,C2	60	26	385,225	19	129	389,175	1	383,871
R17,F05,C2	60	11	782,334	5	68	816,318	1	741,136
R17,F10,C2	60	15	1,142,773	7	118	1,242,485	23	1,092,510
R17,F01,C8	60	25	535,475	38	178	551,247	17	530,366
R17,F05,C8	60	9	1,299,937	11	70	1,326,098	116	1,231,700
R17,F10,C8	60	7	2,223,244	5	61	2,308,072	72	1,999,950
R18,F01,C1	60	20	866,802	70	148	878,193	6	844,260
R18,F05,C1	60	7	1,609,622	19	60	1,709,883	17	1,575,715
R18,F10,C1	60	9	2,380,708	10	38	2,544,101	2	2,240,660
R18,F01,C2	60	11	966,838	51	129	974,302	108	941,763
R18,F05,C2	60	3	1,955,396	4	18	2,003,180	27	1,883,870
R18,F10,C2	60	4	3,157,951	5	26	3,297,777	75	2,792,870
R18,F01,C8	60	5	1,572,109	12	26	1,585,390	55	1,540,690
R18,F05,C8	60	5	4,312,752	7	21	4,572,846	61	4,039,410
R18,F10,C8	60	3	6,899,618	4	7	7,007,133	4	6,608,210

Table 4: Computational results on R instances

Instances	TS	PR	Intensification
R	.0090	.0201	.0052
C	.0114	.0061	.0035

Table 5: Average ratios of standard deviations to mean

Instances	O. Int./TS	C. Int./TS
54 R	-1.86%	-3.94%
24 C	-0.98%	-2.26%

Table 6: Improvements from intensification compared with solutions generated by the tabu-search procedure

Pedersen et al. (2009) and CPLEX after 1 and 5 hours, respectively, while Column "TS-PR/LB.CPLEX" reports the average gap with respect to the best CPLEX lower bound obtained after 10 hours. Table 7 shows that TS-PR improves on CPLEX run for 1 hour, identifying 34 equal or better solutions for the R instances and 19 better solutions for the C instances. Allowing CPLEX 5 hours of computing times improves slightly the solution for a few instances, but still cannot find feasible solutions for two C instances, while the proposed method identifies 22 better solutions for R instances and 7 for C instances. The excellent performance of the proposed TS-PR meta-heuristic is further underlined by the low gaps with the lower bound identified by CPLEX after 10 hours, which indicates that TS-PR reaches excellent-quality solutions leaving little to be further improved.

Instances	TS-PR/CPLEX 1h		TS-PR/CPLEX 5h		TS-PR/B. CPLEX
	Gap	No.Sol.	Gap	No.Sol.	
54 R	-0.70%	34	0.10%	22	1.61%
24 C	-0.71%	19	0.12%	7	1.82%

Table 7: Comparative performance measures of TS-PR versus CPLEX

We also compared the performance of the algorithm we propose to that of Pedersen et al. (2009). The figures in Table 8 show that TS-PR outperforms the state-of-the-art tabu-search, identifying better solutions for all problem instances with an average improvement of 5.94% for R instances and 4.71% for C instances. Table 8 also displays the performance of the tabu-search phase of TS-PR. The figures clearly show that the proposed tabu-search procedure outperforms the one in Pedersen et al. (2009) by identifying 43 improved solutions for R instances and 22 for C instances. The improvements are significant, of more than 2% on average.

We complete this analysis by examining the behavior of the intensification procedure relative to that of CPLEX. Table 9 displays the number of arcs fixed by the fixing scheme of the procedure. Column "O. Int./O. CPLEX" gives the average ratio of the number of arcs fixed to open during the first phase of the procedure to the number of open

Instances	TS-PR/P-TS		TS/P-TS	
	Gap	No.Sol.	Gap	No.Sol.
54 R	-5.94%	54	-2.05%	43
24 C	-4.71%	24	-2.45%	22

Table 8: Comparison of TS-PR and tabu search procedure in TS-PR with Pedersen et al. (2009)

arcs in the solution found by CPLEX 12.1 in 5 hours. These ratios are high, which indicates that the solver needs to add only a few arcs to satisfy all the constraints of each DBCMND instance. Column "C.Int./Arcs" gives the average ratio of the number of arcs fixed to closed during the second phase of the procedure to the total number of arcs in the instance. As can be seen, the fixing scheme in the second phase closes many arcs, which substantially reduces the size of the search space explored by CPLEX. Column "(C. Int+O. CPLEX)/Arcs" gives the average ratio of the number of arcs fixed to closed in the second phase plus the arcs that are open in the CPLEX solution, to the total number of arcs in the instance. In general, the ratios indicate that the intensification procedure is successful in reducing the search space, so that the successive search performed by CPLEX is computationally effective.

Instances	O. Int./O. CPLEX	C. Int./Arcs	(C. Int.+O.CPLEX)/Arcs
54 R	80%	61%	86%
24 C	82%	80%	94%

Table 9: Ratios of fixed arcs by the intensification phase versus CPLEX

6 Conclusions

We have proposed a matheuristic solution framework that combines tabu-search and path-relinking for the DBCMND problem. We considered a total of 78 instances. The tabu-search phase is competitive with the current state-of-the-art tabu-search algorithm, obtaining 65 improved solutions. We introduced a cycle-based neighborhood structure and a minimum-cost maximum-flow model to satisfy the design-balance constraints for the DBCMND. With the help of the tabu-search and restricted path-relinking phases, the proposed matheuristic found 78 improved solutions compared to those of the state-of-the-art tabu-search, and 63 improved solutions compared to those of CPLEX 12.1 (1 hour). The results are also competitive with those found by CPLEX 12.1 (5 hours) with a small average difference and 29 improved solutions with less computational time on average.

Several directions could be investigated to improve the performance of our algorithm.

An adaptive guiding scheme could force the tabu-search to escape from local optima or to concentrate on the region close to the current solution. We could also extend the algorithm to sparse graphs in which it is not easy to find feasible solutions because of the lack of arcs to add to the candidate solutions.

Acknowledgments

While working on this project, T.G. Crainic was the NSERC Industrial Research Chair in Logistics Management, ESG UQAM. and Adjunct Professor with the Department of Computer Science and Operations Research, Université de Montréal, and the Department of Economics and Business Administration, Molde University College, Norway, while M. Toulouse was Adjunct Professor with the Department of Computer Science and Operations Research, Université de Montréal.

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 Grant programs, and by the Fonds québécois de la recherche sur la nature et les technologies (FQRNT).

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. 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 & Sons, New York, NY, 1997.

C. Barnhart, N. Krishnan, D. Kim, and K. Ware. Network Design for Express Shipment Delivery. *Computational Optimization and Applications*, 21(3):239–262, 2002.

M. Chouman and T. Crainic. A MIP-Tabu Search Hybrid Framework for Multicommodity Capacitated Fixed-Charge Network Design. Technical Report CIRRELT-2010-31, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada, 2010.

M. Christiansen, K. Fagerholt, B. Nygreen, and D. Ronen. Maritime Transportation. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 189–284. North-Holland, Amsterdam, 2007.

J.-F. Cordeau, P. Toth, and D. Vigo. A Survey of Optimization Models for Train Routing and Scheduling. *Transportation Science*, 32(4):380–404, 1998.

T.G. Crainic. Network Design in Freight Transportation. *European Journal of Operational Research*, 122(2):272–288, 2000.

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.

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

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.

F. Glover. Tabu Search – Part I. *ORSA Journal on Computing*, 1(3):190–206, 1989.

F. Glover. Tabu Search – Part II. *ORSA Journal on Computing*, 2(1):4–32, 1990.

F. Glover. A Template for Scatter Search and Path Relinking. In J. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer Verlag, Berlin, 1997.

F. Glover, M. Laguna, and R. Martí. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3):653–684, 2000.

D. Kim, C. Barnhart, K. Ware, and G. Reinhardt. Multimodal Express Package Delivery: A Service Network Design Application. *Transportation Science*, 33(4):391–407, 1999.

T.L. Magnanti and R. Wong. Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, 18(1):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.

Appendix

Table 10 explains the column headers used in the subsequent tables.

Column	Meaning
Instances	Instances used for experiments
TS-PR	Best solutions found by the TS-PR matheuristic, which includes tabu-search, path-relinking, and intensification phases
P-TS	Best solutions found by Pedersen et al. (2009)
TS	Best objective values found by proposed tabu search
PR	Best objective values found by proposed path relinking
MP	Best objective values found by Express-MP in 1 hour Pedersen et al. (2009)
CPLEX 1 h	Best solutions found by CPLEX 12.1 in 1 hour
CPLEX 5 h	Best solutions found by CPLEX 12.1 in 5 hours
B. CPLEX	Best lower bound found by CPLEX 12.1 in 10 hours
X/Y	Comparison of results found by procedure X in percentage improvement with respect to those of procedure Y
Nodes	Number of nodes in each instance
Arcs	Number of arcs in each instance
Com.	Number of commodities in each instance
O. Int.	Average number of arcs fixed to open during intensification
C. Int.	Average number of arcs fixed to closed during intensification
O. CPLEX	Number of open arcs in best solutions found by CPLEX after 5 hours
Int. Open	Best value found by fixing-open intensification scheme
Int. Close	Best value found by fixing-close intensification scheme
Total System Imbalance	Total system imbalance of solution found by CPLEX after 1 hour
Running Time	Running time for each instance
Time CPLEX 5 h	CPLEX CPU time (min); The 300 figure indicates that CPLEX cannot find the optimal solution in 5 hours
No. TS	Average number of solutions found by the tabu-search phase
No. PR	Average number of solutions found by the path-relinking phase

Table 10: Notation used in tables of the Appendix

Table 11 lists the solutions for the instances used in the calibration phase.

Instance	Bound	P-TS	TS	TS-PR
R10,F05,C2	436,073	443,547	445,383	439,244
R12,F10,C2	7,408,996	7,530,870	7,467,136	7,436,420
R13,F01,C8	218,787	223,231	219,105	218,787
R15,F10,C8	9,105,010	9,366,760	9,166,884	9,105,010
C20,230,200,F,L	128,014	146,643	146,445	139,365
R16,F10,C1	309,383	352,681	345,198	340,641
R17,F01,C1	364,784	365,801	366,914	364,995
R18,F05,C1	1,362,596	1,597,610	1,601,402	1,582,820
C30,520,100,V,T	52,622	53,972	53,144	53,032
C100,400,30,F,L	58,316	67,603	69,658	67,103

Table 11: Best solutions for instances used in calibration

Tables 12 and 13 display the characteristics of the R and C instances in terms of number of nodes, arcs, and commodities, as well as cost and capacity ratios (see, e.g., Ghamlouche et al., 2003, for more detailed information). For the C instances, a high or low fixed cost relative to the routing cost is signaled by the letter F or V, respectively, while the letters T and L indicate, respectively, if the problem has a tight or loose capacity given the total demand. For the R instances, the fixed cost ratio is computed as $|\mathcal{P}| \sum_{(i,j)\in\mathcal{A}} f_{ij} / \sum_{p\in\mathcal{P}} \sum_{(i,j)\in\mathcal{A}} c_{ij}^p$, and the three values considered are F01 = 0.01, F05 = 0.05, and F10 = 0.10 corresponding to increasing levels of fixed costs compared to routing costs. The capacity ratio is computed as $|\mathcal{A}| \sum_{p\in\mathcal{P}} w^p / \sum_{(i,j)\in\mathcal{A}} u_{ij}$, and the values considered are C1 = 1, C2 = 2, and C8 = 8, indicating that the total capacity becomes increasingly tight relative to the total demand.

Instance	Nodes	Arcs	Commodities
R13,F01,C1	20	220	40
R14,F01,C1	20	220	100
R15,F01,C1	20	220	200
R16,F01,C1	20	314	40
R17,F01,C1	20	318	100
R18,F01,C1	20	315	200

Table 12: Characteristics of R instances

Tables 14 and 15 display the objective values for each method and the gaps between the proposed algorithm and the other methods for the C instances. Tables 16 and 17 display the same information for the R instances. Negative gaps indicate the proposed method improves over previous ones; some of these values (Columns "TS-PR/P-TS") are considerable. The values in Columns "TS-PR/CPLEX 1 h" and "TS-PR/CPLEX 5 h" indicate that our algorithm is also competitive with the state-of-the-art solver. The maximum difference between the results of the proposed algorithm and CPLEX 5 h is

Instance	Nodes	Arcs	Commodities
C20,230,200,V,L	20	230	200
C20,230,200,F,L	20	230	200
C20,230,200,V,T	20	230	200
C20,230,200,F,T	20	230	200
C20,300,200,V,L	20	300	200
C20,300,200,F,L	20	300	200
C20,300,200,V,T	20	300	200
v C20,300,200,F,T	20	300	200
C30,520,100,V,L	30	520	100
C30,520,100,F,L	30	520	100
C30,520,100,V,T	30	520	100
C30,520,100,F,T	30	520	100
C30,520,400,V,L	30	520	400
C30,520,400,F,L	30	520	400
C30,520,400,V,T	30	520	400
C30,520,400,F,T	30	520	400
C30,700,100,V,L	30	700	100
C30,700,100,F,L	30	700	100
C30,700,100,V,T	30	700	100
C30,700,100,F,T	30	700	100
C30,700,400,V,L	30	700	400
C30,700,400,F,T	30	700	400
C30,700,400,V,T	30	700	400
C30,700,400,F,T	30	700	400

Table 13: Characteristics of C instances

less than 1.5%, and many values are less than 0.5%. This quality is obtained in less computational time than the MIP solver.

Instance	P-TS	CPLEX 1 h	CPLEX 5 h	B. CPLEX	TS-PR
C20,230,200,V,L	102,919	98,976	97,847	96,038	97,274
C20,230,200,F,L	150,764	141,689	140,843	136,276	139,395
C20,230,200,V,T	103,371	101,696	100,558	100,209	100,720
C20,230,200,F,T	149,942	141,671	140,108	136,204	138,962
C20,300,200,V,L	82,533	78,168	77,742	76,459	77,584
C20,300,200,F,L	128,757	122,164	119,823	116,773	119,987
C20,300,200,V,T	78,571	76,602	76,208	76,190	76,450
C20,300,200,F,T	116,338	114,816	111,475	109,012	111,776
C30,520,100,V,L	55,981	54,683	54,683	54,677	54,783
C30,520,100,F,L	104,533	101,346	99,900	95,090	100,098
C30,520,100,V,T	54,493	53,041	53,023	52,998	53,035
C30,520,100,F,T	105,167	102,090	101,271	98,653	101,412
C30,520,400,V,L	119,735	115,167	114,646	113,769	115,528
C30,520,400,F,L	162,360	153,311	153,311	150,144	153,409
C30,520,400,V,T	120,421	n/a	n/a	116,111	117,226
C30,520,400,F,T	161,978	n/a	n/a	152,725	155,906
C30,700,100,V,L	49,902	48,855	48,693	48,688	48,807
C30,700,100,F,L	63,889	61,846	61,362	60,236	61,408
C30,700,100,V,T	48,202	46,792	46,750	46,582	46,812
C30,700,100,F,T	58,204	56,251	56,287	55,732	56,237
C30,700,400,V,L	103,932	101,237	99,905	98,323	100,589
C30,700,400,F,T	157,043	n/a	139,495	133,762	141,037
C30,700,400,V,T	103,085	n/a	97,800	96,013	97,875
C30,700,400,F,T	141,917	133,194	133,194	130,066	133,686

Table 14: Best solutions for C instances

Tables 18 and 19 compare the performance of the proposed tabu search to that of the other methods. The tabu-search phase finds feasible solutions for all instances, and improves over the state-of-the-art tabu-search method 22 (of 24) C instances and 47 (of 54) R instances. The largest improvements are obtained for instances with high cost ratios ("F10" for R instances and "F" for C instances) because the feasibility phase always adds the arcs with the smallest total cost when satisfying the design-balance constraints, which has a more important impact when fixed costs are high.

Tables 20 and 21 give the number of arcs fixed by each type of intensification and CPLEX after 5 hours, as well as comparative ratios. Column 5 compares the number of open arcs by the intensification procedure and CPLEX. The high values in this column indicate that the intensification phase yields good results, the solver having to add only a small number of arcs to satisfy all the constraints. Column 6 displays the ratio of

Instance	TS-PR/ P-TS	TS-PR/ CPLEX 1 h	TS-PR/ CPLEX 5 h	TS-PR/ B.CPLEX
C20,230,200,V,L	-5.80%	-1.75%	-0.59%	1.27%
C20,230,200,F,L	-8.16%	-1.65%	-1.04%	2.24%
C20,230,200,V,T	-2.63%	-0.97%	0.16%	0.51%
C20,230,200,F,T	-7.90%	-1.95%	-0.82%	1.98%
C20,300,200,V,L	-6.38%	-0.75%	-0.20%	1.45%
C20,300,200,F,L	-7.31%	-1.81%	0.14%	2.68%
C20,300,200,V,T	-2.77%	-0.20%	0.32%	0.34%
C20,300,200,F,T	-4.08%	-2.72%	0.27%	2.47%
C30,520,100,V,L	-2.19%	0.18%	0.18%	0.19%
C30,520,100,F,L	-4.43%	-1.25%	0.20%	5.00%
C30,520,100,V,T	-2.75%	-0.01%	0.02%	0.07%
C30,520,100,F,T	-3.70%	-0.67%	0.14%	2.72%
C30,520,400,V,L	-3.64%	0.31%	0.76%	1.52%
C30,520,400,F,L	-5.83%	0.06%	0.06%	2.13%
C30,520,400,V,T	-2.73%	n/a	n/a	0.95%
C30,520,400,F,T	-3.89%	n/a	n/a	2.04%
C30,700,100,V,L	-2.24%	-0.10%	0.23%	0.24%
C30,700,100,F,L	-4.04%	-0.71%	0.07%	1.91%
C30,700,100,V,T	-2.97%	0.04%	0.13%	0.49%
C30,700,100,F,T	-3.50%	-0.02%	-0.09%	0.90%
C30,700,400,V,L	-3.32%	-0.64%	0.68%	2.25%
C30,700,400,F,T	-11.35%	n/a	1.09%	5.16%
C30,700,400,V,T	-5.32%	n/a	0.08%	1.90%
C30,700,400,F,T	-4.71%	-0.71%	0.10%	1.82%

Table 15: TS-PR improvement with respect to other methods for C instances

Instance	P-TS	CPLEX 1 h	CPLEX 5 h	B.CPLEX	TS-PR
R13,F01,C1	147,837	147,349	147,349	147,349	147,349
R13,F05,C1	281,668	277,891	277,891	277,891	279,389
R13,F10,C1	404,434	385,396	385,396	385,396	385,396
R13,F01,C2	159,852	155,887	155,887	155,887	156,616
R13,F05,C2	311,209	295,180	295,180	295,180	295,180
R13,F10,C2	470,034	444,545	433,117	431,140	434,383
R13,F01,C8	225,339	218,787	218,787	218,787	218,787
R13,F05,C8	512,027	491,603	491,560	486,754	492,959
R13,F10,C8	875,984	789,479	782,049	772,790	791,213
R14,F01,C1	431,562	422,709	422,709	422,709	422,709
R14,F05,C1	811,102	807,553	784,884	784,626	784,626
R14,F10,C1	1,193,950	1,199,250	1,132,900	1,094,083	1,137,820
R14,F01,C2	465,762	452,591	452,591	452,591	453,434
R14,F05,C2	942,678	892,534	884,234	875,645	891,138
R14,F10,C2	1,401,880	1,320,570	1,314,460	1,256,060	1,307,770
R14,F01,C8	720,882	702,781	702,614	702,614	702,614
R14,F05,C8	1,795,650	1,747,030	1,691,770	1,671,457	1,693,240
R14,F10,C8	2,997,290	2,767,180	2,758,650	2,735,090	2,769,360
R15,F01,C1	1,039,440	1,017,740	1,017,740	1,017,740	1,017,740
R15,F05,C1	2,170,310	2,011,860	2,011,860	1,976,249	2,055,803
R15,F10,C1	3,194,270	3,011,740	2,980,870	2,850,055	2,971,500
R15,F01,C2	1,205,790	1,176,050	1,174,960	1,174,517	1,174,520
R15,F05,C2	2,698,680	2,607,500	2,564,840	2,508,981	2,561,060
R15,F10,C2	4,447,950	4,422,350	4,030,490	3,887,960	4,045,030
R15,F01,C8	2,472,860	2,401,180	2,401,115	2,401,110	2,408,210
R15,F05,C8	6,067,350	5,795,320	5,795,320	5,795,320	5,796,510
R15,F10,C8	10,263,600	9,105,010	9,105,010	9,105,010	9,129,360
R16,F01,C1	142,692	140,082	140,082	140,082	140,082
R16,F05,C1	261,755	248,703	248,703	248,703	248,703
R16,F10,C1	374,819	344,446	340,641	340,641	350,958
R16,F01,C2	145,266	142,381	142,381	142,381	142,605
R16,F05,C2	277,307	260,993	259,313	259,313	260,822
R16,F10,C2	391,386	361,626	365,001	361,626	368,572
R16,F01,C8	187,176	179,639	179,639	179,639	180,228
R16,F05,C8	423,320	391,101	390,549	380,855	388,180
R16,F10,C8	649,121	599,456	596,660	583,389	598,835
R17,F01,C1	374,016	364,784	364,784	364,784	365,788
R17,F05,C1	718,135	686,722	686,234	660,755	676,528
R17,F10,C1	1,041,450	989,809	968,207	908,867	966,116
R17,F01,C2	393,608	382,593	382,593	382,593	384,579
R17,F05,C2	786,198	755,416	741,984	734,117	741,744
R17,F10,C2	1,162,290	1,113,030	1,086,710	1,034,154	1,086,640
R17,F01,C8	539,817	530,435	529,350	525,189	529,876
R17,F05,C8	1,348,750	1,232,750	1,224,770	1,204,567	1,230,910
R17,F10,C8	2,227,780	2,043,920	2,008,220	1,958,676	1,999,950
R18,F01,C1	864,425	845,718	846,857	842,109	844,260
R18,F05,C1	1,640,200	1,606,880	1,597,740	1,556,693	1,588,890
R18,F10,C1	2,399,230	2,359,170	2,233,090	2,149,024	2,264,470
R18,F01,C2	962,402	942,884	941,635	934,187	944,708
R18,F05,C2	1,958,150	1,908,250	1,885,570	1,833,797	1,883,870
R18,F10,C2	2,986,000	2,813,750	2,813,750	2,709,098	2,806,020
R18,F01,C8	1,617,320	1,563,820	1,536,600	1,520,032	1,542,500
R18,F05,C8	4,268,580	4,039,340	3,993,820	3,934,504	4,039,410
R18,F10,C8	7,440,780	6,639,848	6,578,230	6,503,621	6,603,500

Table 16: Best solutions for R instances

Instance	TS-PR/ P-TS	TS-PR/ CPLEX 1 h	TS-PR/ CPLEX 5 h	TS-PR/ B.CPLEX
R13,F01,C1	-0.33%	0.00%	0.00%	0.00%
R13,F05,C1	-1.36%	0.00%	0.00%	0.00%
R13,F10,C1	-4.94%	0.00%	0.00%	0.00%
R13,F01,C2	-2.54%	0.00%	0.00%	0.00%
R13,F05,C2	-5.26%	0.16%	0.16%	0.16%
R13,F10,C2	-7.62%	-1.78%	0.84%	1.29%
R13,F01,C8	-2.99%	0.00%	0.00%	0.00%
R13,F05,C8	-3.87%	0.28%	0.28%	1.26%
R13,F10,C8	-10.93%	0.02%	0.96%	2.13%
R14,F01,C1	-1.77%	0.31%	0.31%	0.31%
R14,F05,C1	-3.37%	-2.92%	-0.03%	0.00%
R14,F10,C1	-5.48%	-5.95%	-0.09%	3.34%
R14,F01,C2	-2.58%	0.32%	0.32%	0.32%
R14,F05,C2	-6.75%	-1.07%	-0.13%	0.84%
R14,F10,C2	-7.17%	-0.96%	-0.49%	3.97%
R14,F01,C8	-2.51%	0.07%	0.09%	0.09%
R14,F05,C8	-5.93%	-3.06%	0.20%	1.40%
R14,F10,C8	-8.69%	-0.35%	-0.04%	0.82%
R15,F01,C1	-1.97%	0.16%	0.16%	0.16%
R15,F05,C1	-7.59%	0.26%	0.26%	2.03%
R15,F10,C1	-6.99%	-0.88%	0.16%	4.54%
R15,F01,C2	-2.66%	-0.13%	-0.04%	0.00%
R15,F05,C2	-4.93%	-1.38%	0.27%	2.45%
R15,F10,C2	-10.72%	-10.08%	-0.33%	3.22%
R15,F01,C8	-2.68%	0.29%	0.29%	0.29%
R15,F05,C8	-4.67%	0.02%	0.02%	0.02%
R15,F10,C8	-12.42%	0.27%	0.27%	0.27%
R16,F01,C1	-1.86%	0.00%	0.00%	0.00%
R16,F05,C1	-5.25%	0.00%	0.00%	0.00%
R16,F10,C1	-8.57%	0.23%	1.33%	1.33%
R16,F01,C2	-1.87%	0.16%	0.16%	0.16%
R16,F05,C2	-5.87%	0.36%	1.00%	1.00%
R16,F10,C2	-7.52%	-0.28%	0.65%	0.65%
R16,F01,C8	-3.91%	0.27%	0.27%	0.27%
R16,F05,C8	-8.05%	0.18%	0.32%	2.79%
R16,F10,C8	-7.39%	0.82%	1.29%	3.48%
R17,F01,C1	-2.05%	0.47%	0.47%	0.47%
R17,F05,C1	-6.15%	-1.51%	-1.43%	2.33%
R17,F10,C1	-9.28%	-3.86%	-1.59%	4.63%
R17,F01,C2	-2.54%	0.33%	0.33%	0.33%
R17,F05,C2	-6.08%	-1.93%	-0.11%	0.95%
R17,F10,C2	-6.39%	-1.88%	0.53%	5.34%
R17,F01,C8	-1.78%	-0.01%	0.19%	0.98%
R17,F05,C8	-9.50%	-0.09%	0.56%	2.20%
R17,F10,C8	-11.39%	-2.20%	-0.41%	2.06%
R18,F01,C1	-2.39%	-0.17%	-0.31%	0.25%
R18,F05,C1	-4.09%	-1.98%	-1.40%	1.21%
R18,F10,C1	-7.08%	-5.29%	0.34%	4.09%
R18,F01,C2	-2.19%	-0.12%	0.01%	0.80%
R18,F05,C2	-3.94%	-1.29%	-0.09%	2.66%
R18,F10,C2	-6.92%	-0.75%	-0.75%	3.00%
R18,F01,C8	-4.97%	-1.50%	0.27%	1.34%
R18,F05,C8	-5.67%	0.00%	1.13%	2.60%
R18,F10,C8	-12.60%	-0.48%	0.45%	1.58%

Table 17: TS-PR improvement with respect to other methods for R instances

Instance	P-TS	CPLEX 1 h	TS	TS/P-TS	TS/CPLEX 1 h
C20,230,200,V,L	102,919	98,976	100,708	-2.20%	1.72%
C20,230,200,F,L	150,764	141,689	145,958	-3.29%	2.92%
C20,230,200,V,T	103,371	101,696	102,906	-0.45%	1.18%
C20,230,200,F,T	149,942	141,671	142,038	-5.56%	0.26%
C20,300,200,V,L	82,533	78,168	80,166	-2.95%	2.49%
C20,300,200,F,L	128,757	122,164	124,846	-3.13%	2.15%
C20,300,200,V,T	78,571	76,602	77,692	-1.13%	1.40%
C20,300,200,F,T	116,338	114,816	119,384	2.55%	3.83%
C30,520,100,V,L	55,981	54,683	55,002	-1.78%	0.58%
C30,520,100,F,L	104,533	101,346	102,735	-1.75%	1.35%
C30,520,100,V,T	54,493	53,041	53,313	-2.21%	0.51%
C30,520,100,F,T	105,167	102,090	102,484	-2.62%	0.38%
C30,520,400,V,L	119,735	115,167	118,627	-0.93%	2.92%
C30,520,400,F,L	162,360	153,311	155,270	-4.57%	1.26%
C30,520,400,V,T	120,421	n/a	119,795	-0.52%	n/a
C30,520,400,F,T	161,978	n/a	156,694	-3.37%	n/a
C30,700,100,V,L	49,902	48,855	48,879	-2.09%	0.05%
C30,700,100,F,L	63,889	61,846	61,745	-3.47%	-0.16%
C30,700,100,V,T	48,202	46,792	47,141	-2.25%	0.74%
C30,700,100,F,T	58,204	56,251	56,810	-2.45%	0.98%
C30,700,400,V,L	103,932	101,237	101,423	-2.47%	0.18%
C30,700,400,F,T	157,043	n/a	141,037	-11.35%	n/a
C30,700,400,V,T	103,085	n/a	108,481	4.97%	n/a
C30,700,400,F,T	141,917	133,194	134,320	-5.66%	0.84%

Table 18: Improvement achieved by the tabu-search procedure on C instances

Instance	P-TS	CPLEX 1 h	TS	TS/P-TS	TS/CPLEX 1 h
R13,F01,C1	147,837	147,349	148,425	0.40%	0.72%
R13,F05,C1	281,668	277,891	289,535	2.72%	4.02%
R13,F10,C1	404,434	385,396	403,490	-0.23%	4.48%
R13,F01,C2	159,852	155,887	157,743	-1.34%	1.18%
R13,F05,C2	311,209	295,180	306,783	-1.44%	3.78%
R13,F10,C2	470,034	444,545	456,269	-3.02%	2.57%
R13,F01,C8	225,339	218,787	219,105	-2.85%	0.15%
R13,F05,C8	512,027	491,603	500,403	-2.32%	1.76%
R13,F10,C8	875,984	789,479	813,589	-7.67%	2.96%
R14,F01,C1	431,562	422,709	437,836	1.43%	3.45%
R14,F05,C1	811,102	807,553	853,765	5.00%	5.41%
R14,F10,C1	1,193,950	1,199,250	1,214,782	1.71%	1.28%
R14,F01,C2	465,762	452,591	455,609	-2.23%	0.66%
R14,F05,C2	942,678	892,534	914,839	-3.04%	2.44%
R14,F10,C2	1,401,880	1,320,570	1,378,765	-1.68%	4.22%
R14,F01,C8	720,882	702,781	714,841	-0.85%	1.69%
R14,F05,C8	1,795,650	1,747,030	1,743,116	-3.01%	-0.22%
R14,F10,C8	2,997,290	2,767,180	2,874,717	-4.26%	3.74%
R15,F01,C1	1,039,440	1,017,740	1,033,662	-0.56%	1.54%
R15,F05,C1	2,170,310	2,011,860	2,156,433	-0.64%	6.70%
R15,F10,C1	3,194,270	3,011,740	3,362,675	5.01%	10.44%
R15,F01,C2	1,205,790	1,176,050	1,177,198	-2.43%	0.10%
R15,F05,C2	2,698,680	2,607,500	2,785,075	3.10%	6.38%
R15,F10,C2	4,447,950	4,422,350	4,430,712	-0.39%	0.19%
R15,F01,C8	2,472,860	2,401,180	2,408,210	-2.68%	0.29%
R15,F05,C8	6,067,350	5,795,320	5,874,704	-3.28%	1.35%
R15,F10,C8	10,263,600	9,105,010	9,205,777	-11.49%	1.09%
R16,F01,C1	142,692	140,082	141,352	-0.95%	0.90%
R16,F05,C1	261,755	248,703	260,685	-0.41%	4.60%
R16,F10,C1	374,819	344,446	371,854	-0.80%	7.37%
R16,F01,C2	145,266	142,381	143,345	-1.34%	0.67%
R16,F05,C2	277,307	260,993	264,345	-4.90%	1.27%
R16,F10,C2	391,386	361,626	375,258	-4.30%	2.73%
R16,F01,C8	187,176	179,639	181,350	-3.21%	0.94%
R16,F05,C8	423,320	391,101	398,082	-6.34%	1.75%
R16,F10,C8	649,121	599,456	629,107	-3.18%	4.71%
R17,F01,C1	374,016	364,784	372,914	-0.30%	2.18%
R17,F05,C1	718,135	686,722	695,550	-3.25%	1.27%
R17,F10,C1	1,041,450	989,809	1,037,684	-0.36%	4.61%
R17,F01,C2	393,608	382,593	385,225	-2.18%	0.68%
R17,F05,C2	786,198	755,416	782,334	-0.49%	3.44%
R17,F10,C2	1,162,290	1,113,030	1,142,773	-1.71%	2.60%
R17,F01,C8	539,817	530,435	535,475	-0.81%	0.94%
R17,F05,C8	1,348,750	1,232,750	1,299,937	-3.76%	5.17%
R17,F10,C8	2,227,780	2,043,920	2,223,244	-0.20%	8.07%
R18,F01,C1	864,425	845,718	866,802	0.27%	2.43%
R18,F05,C1	1,640,200	1,606,880	1,609,622	-1.90%	0.17%
R18,F10,C1	2,399,230	2,359,170	2,380,708	-0.78%	0.90%
R18,F01,C2	962,402	942,884	966,838	0.46%	2.48%
R18,F05,C2	1,958,150	1,908,250	1,955,396	-0.14%	2.41%
R18,F10,C2	2,986,000	2,813,750	3,157,951	5.45%	10.90%
R18,F01,C8	1,617,320	1,563,820	1,572,109	-2.88%	0.53%
R18,F05,C8	4,268,580	4,039,340	4,312,752	1.02%	6.34%
R18,F10,C8	7,440,780	6,639,848	6,899,618	-7.84%	3.76%

Table 19: Improvement achieved by the tabu-search procedure on R instances

the number of arcs fixed to closed and the number of arcs in the instance. When the number of arcs is large, closing unpromising arcs helps reduce the search space and the running time. We note that there are some instances for which the entries in this column column are relatively low (about 10% to 40%). These are instances for which the number of arcs in the best solutions found by CPLEX is generally quite large (e.g., instances "R14,F01,C8" and "R15,F01,C8") compared to the number of arcs in these instances, which indicates that most arcs are needed in the design. The values in the last two columns are the ratios of the open-intensification and closed-intensification schemes compared to the tabu-search methods indicating the good behaviour of the procedures.

Instance	O.Int.	C.Int.	O.CPLEX	O.Int./ O.CPLEX	C.Int./ Arcs	Int. Open/ TS	Int. Close/ TS
C20,230,200,V,L	43	129	57	75%	56%	-2.57%	-3.53%
C20,230,200,F,L	38	153	46	83%	67%	-1.61%	-4.71%
C20,230,200,V,T	48	139	58	83%	60%	-1.20%	-2.17%
C20,230,200,F,T	53	142	62	85%	62%	-1.13%	-2.21%
C20,300,200,V,L	60	193	64	94%	64%	-1.15%	-3.33%
C20,300,200,F,L	53	197	62	85%	66%	-3.10%	-4.05%
C20,300,200,V,T	62	192	71	87%	64%	-0.45%	-1.62%
C20,300,200,F,T	57	183	65	88%	61%	-4.40%	-6.81%
C30,520,100,V,L	69	387	91	76%	74%	-0.26%	-0.40%
C30,520,100,F,L	61	395	75	81%	76%	-1.37%	-2.63%
C30,520,100,V,T	103	365	111	93%	70%	-0.08%	-0.52%
C30,520,100,F,T	79	390	91	87%	75%	0.43%	-1.06%
C30,520,400,V,L	124	339	127	98%	65%	-1.33%	-2.68%
C30,520,400,F,L	171	309	113	104%	59%	-0.67%	-1.21%
C30,520,400,V,T	152	324	n/a	n/a	62%	-0.07%	-2.19%
C30,520,400,F,T	227	254	n/a	n/a	49%	-0.35%	-0.51%
C30,700,100,V,L	78	575	84	93%	82%	0.00%	-0.15%
C30,700,100,F,L	62	577	68	91%	82%	-0.18%	-0.55%
C30,700,100,V,T	94	544	104	90%	78%	-0.37%	-0.70%
C30,700,100,F,T	76	540	98	78%	77%	-1.02%	-1.02%
C30,700,400,V,L	105	554	115	91%	79%	-0.73%	-0.83%
C30,700,400,F,T	104	502	113	92%	72%	0.00%	0.00%
C30,700,400,V,T	131	511	138	95%	73%	-1.83%	-10.84%
C30,700,400,F,T	132	536	128	103%	77%	-0.98%	-2.26%

Table 20: Open and closed arc statistics for C instances

Tables 22 and 23 display information relating to the execution of the proposed procedures. The second column gives the system imbalance for the solution obtained by the initialization step, and the third column reports the total running time. The last two columns give the number of solutions found by tabu-search and path-relinking phases.

Instance	O.Int.	C.Int.	O.CPLEX	O.Int./ O.CPLEX	C.Int./ Arcs	Int. Open/ TS	Int. Close/ TS
R13,F01,C1	39	162	45	87%	74%	-0.37%	-0.73%
R13,F05,C1	20	160	34	59%	73%	-1.05%	-4.19%
R13,F10,C1	18	151	31	58%	69%	-3.46%	-4.69%
R13,F01,C2	34	149	44	77%	68%	-0.31%	-1.19%
R13,F05,C2	21	158	37	57%	72%	-2.08%	-3.76%
R13,F10,C2	23	162	36	64%	74%	-4.13%	-4.46%
R13,F01,C8	71	101	81	88%	46%	-0.10%	-0.15%
R13,F05,C8	48	126	68	71%	57%	-0.91%	-1.51%
R13,F10,C8	49	124	66	74%	56%	-2.55%	-3.03%
R14,F01,C1	47	125	59	80%	57%	-0.86%	-3.25%
R14,F05,C1	29	141	42	69%	64%	-2.32%	-8.81%
R14,F10,C1	26	161	35	74%	73%	-2.95%	-7.32%
R14,F01,C2	54	115	63	86%	52%	-0.35%	-0.35%
R14,F05,C2	36	146	50	72%	66%	-1.14%	-3.60%
R14,F10,C2	31	155	49	63%	70%	-1.57%	-5.41%
R14,F01,C8	114	22	126	90%	10%	-1.01%	-1.65%
R14,F05,C8	81	93	97	84%	42%	0.90%	-2.83%
R14,F10,C8	65	105	90	72%	48%	-1.70%	-4.24%
R15,F01,C1	63	111	67	94%	50%	-0.42%	-1.40%
R15,F05,C1	48	75	53	91%	34%	-2.84%	-6.90%
R15,F10,C1	37	144	49	76%	65%	-10.27%	-12.63%
R15,F01,C2	72	118	88	82%	54%	-0.15%	-0.23%
R15,F05,C2	65	65	75	87%	30%	-4.81%	-8.29%
R15,F10,C2	71	63	77	92%	29%	-4.03%	-10.29%
R15,F01,C8	159	35	175	91%	16%	0.00%	0.00%
R15,F05,C8	123	39	125	98%	18%	-0.17%	-1.35%
R15,F10,C8	106	81	110	96%	37%	-0.34%	-0.84%
R16,F01,C1	32	245	41	78%	78%	-0.46%	-0.91%
R16,F05,C1	19	219	31	61%	70%	-3.45%	-4.82%
R16,F10,C1	18	250	29	62%	80%	0.00%	-7.71%
R16,F01,C2	36	258	41	88%	82%	-0.16%	-0.52%
R16,F05,C2	22	254	30	73%	81%	-0.66%	-0.92%
R16,F10,C2	24	227	34	71%	72%	-1.81%	-3.09%
R16,F01,C8	50	226	69	72%	72%	-0.09%	-0.68%
R16,F05,C8	37	209	60	62%	67%	-1.36%	-1.60%
R16,F10,C8	36	231	58	62%	74%	-3.73%	-4.08%
R17,F01,C1	40	235	47	85%	74%	-0.29%	-1.75%
R17,F05,C1	28	256	39	72%	81%	-2.81%	-2.81%
R17,F10,C1	18	253	36	50%	80%	-5.24%	-8.89%
R17,F01,C2	50	231	57	88%	73%	0.00%	-0.35%
R17,F05,C2	32	251	43	74%	79%	-3.75%	-5.56%
R17,F10,C2	35	244	42	83%	77%	-3.77%	-4.60%
R17,F01,C8	84	181	99	85%	57%	-0.55%	-0.96%
R17,F05,C8	67	203	90	74%	64%	-3.06%	-5.54%
R17,F10,C8	57	208	88	65%	65%	-3.53%	-11.16%
R18,F01,C1	54	183	65	83%	58%	-0.97%	-2.67%
R18,F05,C1	47	226	45	104%	72%	-2.15%	-2.15%
R18,F10,C1	44	211	40	110%	67%	-1.37%	-6.25%
R18,F01,C2	57	200	75	76%	63%	-1.91%	-2.66%
R18,F05,C2	62	238	60	103%	76%	-1.34%	-3.80%
R18,F10,C2	51	246	58	88%	78%	-8.36%	-13.07%
R18,F01,C8	158	124	160	99%	39%	-0.32%	-2.04%
R18,F05,C8	126	157	116	109%	50%	0.00%	-6.77%
R18,F10,C8	108	170	99	109%	54%	-1.86%	-3.94%

Table 21: Open and closed arc statistics for R instances

Instance	System Imbalance	Running Time	Time CPLEX 5 hours	No. TS	No. PR
C20,230,200,V,L	18	91	300	10	85
C20,230,200,F,L	8	92	300	6	40
C20,230,200,V,T	14	78	300	10	43
C20,230,200,F,T	12	84	300	7	33
C20,300,200,V,L	22	162	300	15	95
C20,300,200,F,L	20	110	300	9	47
C20,300,200,V,T	10	90	300	13	68
C20,300,200,F,T	16	141	300	11	88
C30,520,100,V,L	26	85	300	29	185
C30,520,100,F,L	22	118	300	21	187
C30,520,100,V,T	30	103	300	16	166
C30,520,100,F,T	22	165	300	13	109
C30,520,400,V,L	16	221	300	17	157
C30,520,400,F,L	28	146	300	22	110
C30,520,400,V,T	24	210	n/a	19	127
C30,520,400,F,T	38	136	n/a	26	123
C30,700,100,V,L	22	74	67	25	91
C30,700,100,F,L	16	153	300	23	146
C30,700,100,V,T	36	135	300	24	155
C30,700,100,F,T	22	152	300	29	225
C30,700,400,V,L	24	136	300	35	125
C30,700,400,F,T	26	80	300	21	185
C30,700,400,V,T	32	222	300	14	116
C30,700,400,F,T	28	130	300	13	60

Table 22: Statistical information execution C instances

Instance	System Imbalance	Running Time	Time CPLEX 5 hours	No. TS	No. PR
R13,F01,C1	18	62	1	16	78
R13,F05,C1	12	63	54	19	113
R13,F10,C1	8	63	49	15	115
R13,F01,C2	18	62	1	18	80
R13,F05,C2	12	63	105	18	128
R13,F10,C2	16	63	300	13	93
R13,F01,C8	28	62	20	18	82
R13,F05,C8	28	71	300	11	98
R13,F10,C8	22	63	300	18	136
R14,F01,C1	24	69	15	23	140
R14,F05,C1	18	65	300	16	121
R14,F10,C1	14	73	300	6	28
R14,F01,C2	16	65	7	14	81
R14,F05,C2	16	95	300	13	109
R14,F10,C2	16	85	300	9	57
R14,F01,C8	34	107	290	16	71
R14,F05,C8	18	108	300	9	39
R14,F10,C8	28	169	300	10	126
R15,F01,C1	20	67	45	6	31
R15,F05,C1	14	114	300	9	60
R15,F10,C1	10	125	300	4	18
R15,F01,C2	20	87	300	12	66
R15,F05,C2	20	146	300	3	26
R15,F10,C2	18	170	300	5	44
R15,F01,C8	34	117	300	4	10
R15,F05,C8	18	149	17	7	29
R15,F10,C8	34	85	5	7	28
R16,F01,C1	6	62	1	14	54
R16,F05,C1	10	63	126	23	96
R16,F10,C1	12	63	300	11	81
R16,F01,C2	14	64	1	22	103
R16,F05,C2	18	63	100	14	84
R16,F10,C2	18	67	300	12	105
R16,F01,C8	32	66	73	31	146
R16,F05,C8	32	64	300	15	119
R16,F10,C8	24	73	300	18	142
R17,F01,C1	14	71	26	15	83
R17,F05,C1	8	133	300	13	121
R17,F10,C1	18	131	300	13	123
R17,F01,C2	20	80	23	26	129
R17,F05,C2	18	66	300	11	68
R17,F10,C2	20	90	300	15	118
R17,F01,C8	34	115	300	25	178
R17,F05,C8	32	187	300	9	70
R17,F10,C8	32	137	300	7	61
R18,F01,C1	14	136	300	20	148
R18,F05,C1	10	96	300	7	60
R18,F10,C1	10	72	300	9	38
R18,F01,C2	24	219	300	11	129
R18,F05,C2	12	91	300	3	18
R18,F10,C2	12	140	300	4	26
R18,F01,C8	38	127	300	5	26
R18,F05,C8	40	128	300	5	21
R18,F10,C8	20	68	300	3	7

Table 23: Statistical information execution R instances