Transportation Research Part B 144 (2021) 60–80

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

Decomposition methods for large-scale network expansion problems

Ioannis Fragkos[a,*], Jean-François Cordeau[b], Raf Jans[b]

[a] Rotterdam School of Management, the Netherlands
[b] HEC Montréal and CIRRELT, Canada

ARTICLE INFO

Article history:
Received 1 June 2020
Revised 28 September 2020
Accepted 3 December 2020
Available online 25 December 2020

Keywords:
Network design
Multi-period
Lagrange relaxation
Benders decomposition
Heuristics

ABSTRACT

Network expansion problems are a special class of multi-period network design problems in which arcs can be opened gradually in different time periods but can never be closed. Motivated by practical applications, we focus on cases where demand between origin-destination pairs expands over a discrete time horizon. Arc opening decisions are taken in every period, and once an arc is opened it can be used throughout the remaining horizon to route several commodities. Our model captures a key timing trade-off: the earlier an arc is opened, the more periods it can be used for, but its fixed cost is higher, since it accounts not only for construction but also for maintenance over the remaining horizon. An overview of practical applications indicates that this trade-off is relevant in various settings. For the capacitated variant, we develop an arc-based Lagrange relaxation, combined with local improvement heuristics. For uncapacitated problems, we develop four Benders decomposition formulations and show how taking advantage of the problem structure leads to enhanced algorithmic performance. We then utilize real-world and artificial networks to generate 1080 instances, with which we conduct a computational study. Our results demonstrate the efficiency of our algorithms. Notably, for uncapacitated problems we are able to solve instances with 2.5 million variables to optimality in less than two hours of computing time. Finally, we provide insights into how instance characteristics influence the multi-period structure of solutions.

1. Introduction

Network expansion models represent a variety of problems arising in fields as diverse as road construction (Yang et al., 1998), logistics (Lee and Dong, 2008), energy transport and telecommunications (Minoux, 1989), and railways (Hooghiemstra et al., 1999). These problems exhibit a multi-period structure: given a planning horizon and a demand forecast therein, one needs to decide which arcs to open and when, so that the resulting network can accommodate the typically increasing demand throughout the planning horizon. Concretely, opening an arc implies that this arc can be used to route commodities until the end of the planning horizon. However, the more periods the arc is under operation, the higher the

* Corresponding author.
E-mail addresses: fragkos@rsm.nl (I. Fragkos), raf.jans@hec.ca (R. Jans).

https://doi.org/10.1016/j.trb.2020.12.002
0191-2615/© 2020 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license
(http://creativecommons.org/licenses/by/4.0/)

total fixed cost that is incurred. This fixed opening cost represents the total expense over the remaining horizon, such as construction and maintenance costs for building a track in a rail network.

In this paper, we study an archetypal formulation that captures the key timing trade-off of network expansion decisions. On the one hand, building an arc early on implies a high fixed cost, but the arc can be used to route commodities for a large number of subsequent periods. On the other hand, building an arc later in the horizon is associated with a lower fixed cost, but the number of periods in which the arc can be used is smaller. The objective is then to jointly minimize the arc construction and operating costs over the given planning horizon.

Although such network expansion formulations can provide useful input for strategic and tactical decisions, their very large scale makes them difficult or even impossible to solve with modern mixed-integer programming (MIP) technology. To this end, we exploit their multi-period structure to devise specialized decomposition algorithms for both capacitated and uncapacitated variants. First, we apply arc-based Lagrange relaxation for the capacitated problem. Second, we develop a stand-alone heuristic which we combine with Lagrange relaxation. Third, we apply Benders decomposition to uncapacitated problems, by decomposing the original problem into single-period shortest path subproblems per period and per commodity. We then show how Pareto-optimal Benders cuts can be generated efficiently for our application, and compare the resulting implementations with the novel formulation of Fischetti et al. (2010). Further, we employ our algorithms to analyze how problem characteristics, such as capacity tightness, influence the structure of obtained solutions.

In order to illustrate the computational efficiency of our algorithms and investigate the structural characteristics of multi-period solutions, we generate new instances. Specifically, we utilize three actual shipping networks that were used in Pazour et al. (2010) to investigate the usefulness of designing a high-speed freight rail network in the United States. It is worth noting that Pazour et al. (2010) recognize the mutli-period nature of this problem, but resort to solving a simplified, single-period version. Then, we use a subset of the R instances of Crainic et al. (2001), which have been used in a multitude of other studies (Katayama et al., 2009; Yaghini et al., 2014; Costa et al., 2009). In total, we construct more than 1000 instances, with which we perform extensive computational experiments. First, we show that our heuristics, Lagrange relaxation and Benders decomposition are efficient in finding high-quality solutions within a reasonable amount of time, while their performance scales well in larger problem instances. Notably, we are able to solve to optimality instances with 2.5 million variables in less than two hours of CPU time. Second, we are interested in examining structural characteristics of multi-period problem solutions. We deduct several insights, such as that (i) the majority of the arcs are opened in early periods but instances with too short horizons, tight capacities or low fixed costs may open fewer arcs therein; (ii) an increasing commodity demand implies that routing costs are predominant in later periods, regardless of the timing of arc opening; (iii) because of (ii), high-quality solutions may have low capacity utilization, especially in very sparse networks.

The remainder of the paper is organized as follows. We first review related research in Section 2. Then, Section 3 describes the problem formulation and Section 4 explains the construction of a heuristic tailored to large-scale instances. Section 5 provides details on the Lagrange relaxation and Benders decomposition. Then, Section 6 presents computational results. We conclude by reflecting on future research avenues, reported in Section 7.

2. Literature review

The literature on network design and expansion problems is voluminous. In what follows, we first focus on applications related to capacitated and uncapacitated multi-period problems and then provide an overview of methodological advancements in the larger field of network design problems.

2.1. Applications of multi-period problems

Bärmann et al. (2017) consider the problem of expanding the German railway network, whose demand is anticipated to increase by 50% in the coming two decades. In their setting, investments have to be paid throughout the construction periods but the corresponding capacity becomes available only at the last construction period. Lai and Shih (2013) study a related problem, namely stochastic network expansion for railway capacity planning, in which they minimize a combination of network upgrading costs, expected arc operations costs and unfulfilled demand. Other researchers, such as Blanco et al. (2011) and Marin and Jaramillo (2008), focus on application-specific transportation network expansion models using heuristics, while Petersen and Taylor (2001) develop a decision support system to investigate the economic viability of a new railway in Brazil. Cheung et al. (2001) study the problem of redesigning DHL's network of depots and service centers, determining facility capacities and opening timing by solving a multi-period facility location problem, which can be recast as a network design problem, with each facility converted to an arc. In a similar fashion, Pazour et al. (2010), who consider the design of a high-speed rail network for cargo distribution, note that such a network is likely to be built across multiple periods, and assume an incremental design plan by restricting the total length of the network and by fixing prior line construction decisions.

An important class of problems attempts to incorporate user behavior into the design of multi-period networks. Ukkusuri and Patil (2009) consider a multi-period network design problem with investment decisions, demand elasticity and equilibrium constraints. The authors show that flexibility to stage the investment decision over multiple periods can have a large positive impact on the expected consumer surplus. In another study, Tong et al. (2015) maximize user accessibility across major locations, given a fixed investment budget. Although our models are tangential to these works since our use-cases do not involve user behavior, the key network expansion decisions are similar, and our algorithmic approaches remain relevant.

Systems with ample capacity or systems where the pertinent decision is determining the network topology can be represented by uncapacitated models. Capacity restrictions can oftentimes be tackled by post-processing strategies, as in Pazour et al. (2010), who assume that excess flows are directed from high speed rail to the road network. Yet another interesting application is the infrastructure expansion problem in the coal export supply chain, studied by Kalinowski et al. (2015). A special characteristic of this formulation is that it has a single commodity whose demand increases by one unit each time period. Such papers are representative of incremental optimization, a research stream pioneered by Şeref et al. (2009). Our paper adds to this literature by designing tailored algorithms for multi-commodity variants of such incremental problems.

2.2. Methodology

The literature on network design problems is voluminous, ranging from linear single- commodity flow problems to nonlinear capacitated multi-commodity problems and application-specific variants. In what follows, we provide an overview of formulations and methods relevant to our work.

2.2.1. Uncapacitated network design

In a seminal paper, Balakrishnan et al. (1989) utilize dual ascent methods to solve singe-period uncapacitated network design problems with up to 500 integer and 1.98 million continuous variables. Subsequent works focus mostly on exact approaches, such as Lagrange relaxation-based branch-and-bound (Cruz et al., 1998; Holmberg and Yuan, 1998), branch-and-cut, and Benders decomposition (Randazzo and Luna, 2001; Rahmaniani et al., 2016). To the best of our knowledge, the Lagrange relaxation algorithm of Holmberg and Yuan (1998) is the state-of-the-art exact approach for solving single-period uncapacitated problems. A heuristic used by several researchers is the dynamic slope scaling approach of Kim and Pardalos (1999), which the authors utilized for single-commodity uncapacitated and capacitated network design problems. The main idea of slope scaling is to update the objective function coefficients of the continuous variables dynamically, so that the adjusted linear cost is a locally good approximation of both the linear and fixed costs. This idea was adopted by other authors for problems with richer structures, such as capacitated network design (Crainic et al., 2004) and freight rail transportation (Zhu et al., 2014).

2.2.2. Capacitated network design

In spite of some notable research output on the single-period uncapacitated problem, the largest literature stream has focused on solution methods for its capacitated counterpart. To this end, some authors have conducted polyhedral studies (Atamtürk, 2002; Atamtürk and Rajan, 2002; Bienstock and Günlük, 1996; Raack et al., 2011; Günlük, 1999), while others have used column generation (Zetina et al., 2019) and Lagrange relaxation (Cruz et al., 1998; Tong et al., 2015). Problems that model user behavior have a bi-level structure (Wang et al., 2013; Gao et al., 2005), which can also be formulated as a variational-inequality problem (Luathep et al., 2011). For linear bi-level problems, Fontaine and Minner (2014) have used Benders decomposition to tackle large instances, while Liu and Wang (2015) solve a continuous version with stochastic user equilibrium using range reduction and linear outer approximations. In terms of exact methods, it is interesting to note the state-of-the-art study of Chouman et al. (2017), who develop a custom cutting plane algorithm and separation procedures for five classes of valid inequalities. Their experiments show that their separation algorithms perform better on aggregated formulations, i.e., formulations where commodities with a common origin or destination are aggregated into a single commodity, for instances with a small number of commodities, while their algorithms perform better for instances with a large number of commodities when applied to disaggregated formulations.

Since single-period network design is computationally challenging, most authors have adopted heuristic approaches. Yaghini et al. (2014) use a tabu-search algorithm with a neighborhood induced by families of valid inequalities, while Paraskevopoulos et al. (2016) use scatter and local search alongside new search operators that allow partial rerouting of multiple commodities. The computational experiments show that those two approaches are perhaps the most efficient heuristics at the time of this writing, while Katayama et al. (2009), which is based on capacity scaling using column and row generation, remains competitive.

Finally, most papers that consider multi-period variants utilize heuristics, such as Papadimitriou and Fortz (2015), who propose a rolling horizon heuristic to solve practical problem instances. An exception is Petersen and Taylor (2001) who use dynamic programming to solve one specific instance, whose state space can be reduced significantly. Table 1 shows an overview of the key methodologies used in network design and network expansion problems and some representative references.

2.2.3. Summary of methodological contributions

Although regular (single-period) network design problems remain challenging to solve, the additional challenge of network expansion problems comes from their larger size: even solving the linear programming (LP) relaxation of some instances may require a large amount of CPU time, and separating strong inequalities can be time consuming, even when

Table 1
Key characteristics of methods and corresponding research studies.

Method	Type	Output	Periods	References
Dual ascent	Exact or heuristic	Lower bound	Single	Balakrishnan et al. (1989)
Lagrange relaxation	Exact or heuristic	Lower bound	Single	Cruz et al. (1998), Holmberg and Yuan (1998), Holmberg and Yuan (2000)
Benders decomposition	Exact	Upper & lower bound	Single	Rahmaniani et al. (2016), Randazzo and Luna (2001)
Cutting planes	Exact	Upper & lower bound	Single	Atamtürk (2002), Bienstock and Günlük (1996), Raack et al. (2011)
Slope scaling	Heuristic	Upper bound	Single	Kim and Pardalos (1999), Crainic et al. (2004), Katayama et al. (2009)
Column generation	Exact or heuristic	Lower bound	Single	Frangioni and Gendron (2009), Frangioni and Gendron (2013), Alvelos and de Carvalho (2007)
Tabu search, Local search	Heuristic	Upper bound	Single, Multiple	Yaghini et al. (2014), Chouman et al. (2009), Ghamlouche et al. (2003), Marin and Jaramillo (2008)
Dynamic programming	Exact	Upper bound	Multiple	Petersen and Taylor (2001)
Rolling horizon	Heuristic	Upper bound	Multiple	Papadimitriou and Fortz (2015)
Knapsack decomposition	Heuristic	Upper bound	Multiple	Bärmann et al. (2017)

they are polynomial in the problem's input. Other than their sheer size, it is also the embedded single-period network design structure that remains challenging to solve from a computational perspective: Chouman et al. (2017) report that CPLEX 12 may consume up to 10 h of CPU time without proving optimality on a dataset used extensively in the literature. Our instances are multi-period extensions (up to 80 periods) of such instances. Methods that are effective for network design, such as slope scaling heuristics or custom cutting planes, are also applicable to multi-period problems using straightforward adjustments. However, as the space dimensionality increases, their textbook application is unlikely to be successful, unless one exploits the multi-period structure in place. Such a structure can be exploited by decomposition methods.

In uncapacitated problems, the only decisions that influence forward periods are the arc-opening decisions. Given a fixed set of arc-opening decisions, one needs to determine for each commodity and each period independently the minimum-cost path to route that commodity from its origin to its destination, which is a shortest-path problem. This structure suggests that uncapacitated problems are amenable to Benders decomposition, because fixing the binary decisions gives rise to period- and commodity-specific subproblems, each of which can be solved effectively, and the corresponding cuts are separable by each period and commodity combination.

Capacitated problems have the additional complication that in each period commodities interact through arc capacities. Thus, a Benders approach would generate single-period multi-commodity cost flow problems, whose cuts do not separate commodities within each period, and are likely to be weaker. Although cuts from the uncapacitated variant can be added to improve convergence, the resulting scheme is likely to be inefficient computationally when arcs capacities are tight (Fischetti et al., 2016a). This formulation is more amenable to Lagrange relaxation, where the flow balance constraints are dualized in the objective function and the problem decomposes in a series of single-arc, multi-period, multi-commodity problems that we can solve in polynomial time using a custom algorithm. Although the lower bound obtained by such a Langrange relaxation is equivalent to the LP relaxation bound with tight inequalities, solving the LP is often impossible within 2 h of CPU time, while when the LP is solvable our algorithm is much faster.

In summary, our work offers the following methodological contributions. First, we develop an arc-based Lagrange relaxation algorithm for the capacitated problem, which attains a strong lower bound in a short amount of time, particularly for large instances. The computational efficiency of Lagrange relaxation hinges on solving the resulting arc-specific problem in polynomial time. Second, we develop a stand-alone heuristic which we combine with Lagrange relaxation. Computational experiments show that our heuristic is able to attain feasible solutions within 1% of optimality an order of magnitude faster than a commercial MIP solver, while its performance scales better with problem size. Third, this heuristic is used to warm-start a Lagrange relaxation algorithm, which itself uses the volume algorithm (Barahona and Anbil, 2000) to detect promising areas of the search space. This results in an integrated scheme that combines Lagrange relaxation, the volume algorithm, our stand-alone heuristic and local search heuristics that operate during the Lagrange relaxation loop. Fourth, we develop a Benders decomposition algorithm for uncapacitated problems, which decomposes the original problem into single-period shortest path subproblems per period and per commodity. We show how Pareto-optimal (PO) Benders cuts can be generated efficiently for our application using a series of algorithmic enhancements that allow us to generate a PO Benders cut solving a single LP, and compare the resulting implementations with the novel formulation of Fischetti et al. (2010). Finally, we generate new multi-period instances and conduct extensive computational experiments, which benchmark the effectiveness of each approach and use real networks to analyze the structure of the resulting solutions. We next provide a mathematical formulation of the problem we consider.

3. Problem description and formulation

We consider a network of nodes $\mathcal{N}$ and arcs $\mathcal{A}$ and a set of commodities $\mathcal{K}$ that need to be routed from given origin nodes to given destination nodes during each time period $t \in T$. Each commodity should satisfy a period-dependent demand between its origin and its destination. Routing a commodity through an arc incurs a variable, commodity-specific cost. In addition, routing a commodity through an arc is possible only if this arc is opened in this period, or if it has been opened in an earlier period. Opening an arc in a specific period incurs a fixed cost. We introduce the following notation:

Parameters

- f_{ij}^t, cost of opening arc (i, j) at the start of period t
- c_{ij}^k, cost of sending one unit of commodity k through arc (i, j)
- u_{ij}, capacity of arc (i, j)
- d^{kt}, demand of commodity k in period t
- O_k, D_k, origin and destination of commodity k, respectively
- $b_i^k = 1,\ \text{if } i \equiv O_k; -1,\ \text{if } i \equiv D_k;\ 0,\ \text{otherwise}$

Decision Variables

- x_{ij}^{kt}, fraction of d^{kt} that is directed through arc (i, j)
- y_{ij}^t, $= 1$ if arc (i, j) is opened at the beginning of period t, 0 otherwise

The multi-period network expansion problem (M-NEP) is formulated as follows:

$$\min \sum_{t\in T}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^k d^{kt} x_{ij}^{kt} + \sum_{t\in T}\sum_{(i,j)\in\mathcal{A}} f_{ij}^t y_{ij}^t \qquad [M-NEP] \tag{1}$$

$$\text{s.t} \sum_{j:(i,j)\in\mathcal{A}} x_{ij}^{kt} - \sum_{j:(j,i)\in\mathcal{A}} x_{ji}^{kt} = b_i^k, \qquad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in T, \tag{2}$$

$$\sum_{k\in\mathcal{K}} d^{kt} x_{ij}^{kt} \le u_{ij} \sum_{l=1}^{t} y_{ij}^l, \qquad \forall (i,j) \in \mathcal{A}, t \in T, \tag{3}$$

$$x_{ij}^{kt} \le \min\{1, \frac{u_{ij}}{d^{kt}}\} \sum_{l=1}^{t} y_{ij}^l, \qquad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in T, \tag{4}$$

$$\sum_{t\in T} y_{ij}^t \le 1, \qquad \forall (i,j) \in \mathcal{A}, \tag{5}$$

$$0 \le x_{ij}^{kt} \le 1, \qquad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in T, \tag{6}$$

$$y_{ij}^t \in \{0, 1\}, \qquad \forall (i,j) \in \mathcal{A}, \forall t \in T. \tag{7}$$

The objective function (1) minimizes the costs of routing commodities and opening arcs throughout the horizon. Constraints (2) maintain the balance of each commodity in each node and period. Constraints (3) prevent the total amount of flow that is routed through each arc from exceeding that arc's capacity in each period. If at time t, $y_{ij}^s = 0$, for all $1 \le s \le t$, then no flow is routed through the arc (i, j) during period t. Constraints (4) are redundant but potentially useful, since they strengthen the problem's LP relaxation. Specifically, they are the multi-period counterparts of the "strong" inequalities used in single-period problems to improve the LP relaxation (Gendron and Crainic, 1994). Finally, constraints (5) express that each arc can be opened at most once. Arc capacity expansions can be modeled by considering additional pairs of arcs between nodes. This is important for rail networks, where constructing additional tracks to expand the capacity between stations is commonplace (Bärmann et al., 2017).

To avoid trivial solutions, we assume that opening an arc earlier implies a higher cost, i.e., $f_{ij}^t > f_{ij}^{t+1}, \forall (i,j) \in \mathcal{A}, t \in T$. A special case of the above formulation arises when $u_{ij} \ge \sum_{k\in\mathcal{K}} d^{kt}$ for all $(i, j) \in \mathcal{A}$ and $t \in T$. Then, constraints (3) are redundant, and (4) change to $x_{ij}^{kt} \le \sum_{l=1}^{t} y_{ij}^l, \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}$ and $t \in T$. We will hereafter refer to this case as the multi-period *uncapacitated* network expansion problem (M-UNEP). This variant is interesting in its own right because it has different decomposability features than the capacitated variant.

Our formulation can be seen as an extension of single-period multi-commodity network design problems with time-varying demand. As such, it carries similarities to dynamic capacitated facility location problems (Jena et al., 2015), but also capacitated lot sizing problems which have dynamic demand, fixed setup costs and can be recast as capacitated shortest path problems over appropriate networks (De Araujo et al., 2015). All these problems embed the single-node fixed-charge flow model (Atamtürk, 2001), but a key difference is that the demand in lot sizing problems can be satisfied by current or earlier production, while in facility location and network expansion each demand has to be satisfied by the routing decisions taken in each period.

In principle, network design formulations determine simultaneously the routing and design decisions and attempt to spot a balance between the corresponding costs. When the network design decisions y_{ij} are fixed, the formulation reduces to a multi-commodity flow problem, which itself reduces to a series of shortest-path problems when all arcs have infinite capacity. What is perhaps more striking is the generality of the uncapacitated network design formulation. Magnanti and Wong (1984) observe that the Minimum Spanning Tree (MST), Steiner tree, Travelling Salesman, Vehicle routing (VRP) and Facility location problems arise as special cases of the Network Design formulation. While some of these problems are "easy" to solve from a complexity point of view, such as the MST, others, such as the VRP, are themselves matter of extensive research. What makes network design problems significantly harder from a computational perspective is the addition of arc capacities (Holmberg and Yuan, 2000). Such capacities destroy an important property of the uncapacitated models, namely that there exists an optimal solution in which each commodity is routed through a single path. Indeed, in capacitated problems commodity flows may bifurcate, since they compete among each other for arc capacities, giving rise to a class of optimal solutions whose structure is hard to characterize. An example of a small capacitated network with two periods where we show the optimal solution can be found in Appendix A. Before applying decompositions, we introduce a heuristic that generates high-quality solutions fast.

4. Initial heuristic search

Our heuristic tries to detect arcs that are going to be opened in some period, and then decides when it is best to open each of the identified arcs. Algorithm 1 describes the high-level steps of this select-and-time procedure, hereafter

Algorithm 1 The Select-and-Time (S&T) heuristic procedure.

```
Input: $d^{kt}, f^t_{ij}, c^k_{ij}, u_{ij}, b^k_i$
Output: $y^t_{ij}, x^{kt}_{ij}, v_H$
1: $w^t \leftarrow \frac{\sum_{l=t}^{|T|}\sum_{k\in\mathcal{K}} d^{kl}}{\sum_{l\in T}\sum_{k\in\mathcal{K}} d^{kl}}$; $w^t \leftarrow \frac{w^t}{\sum_{l\in T} w^l}$
2: $\hat{f}_{ij} \leftarrow \sum_{t\in T} w^t f^t_{ij}$        ▷ Create weighted fixed cost ($\hat{f}_{ij}$)
3: $\hat{d}^k \leftarrow d^{k1}\frac{\max_{t\in T} d^{kt}}{\frac{1}{|T|}\sum_{t\in T} d^{kt}}$        ▷ Create inflated demand ($\hat{d}^k$)
4: $\hat{y}_{ij} \leftarrow \textsc{SlvSnPer}(\hat{d}^k, \hat{f}_{ij}, c^k_{ij}, u_{ij}, b^k_i)$        ▷ Solve MIP with $\hat{f}_{ij}$, $\hat{d}^k$; Store $\hat{y}_{ij}$
5: $\hat{d}^k \leftarrow \max_{t\in T} d^{kt}$; $\hat{f}_{ij} \leftarrow 0$        ▷ Take max demand ($\hat{d}^k$), set zero fixed costs ($\hat{f}_{ij}$)
6: $\bar{y}_{ij} \leftarrow \textsc{SlvSnPer}(\hat{d}^k, \hat{f}_{ij}, c^k_{ij}, u_{ij}, b^k_i)$        ▷ Solve LP with $\hat{f}_{ij}$ $\hat{d}^k$; Store arcs ($\bar{y}_{ij}$)
7: $\mathcal{A}^{pot} = \{(i,j)\in\mathcal{A} \mid \hat{y}_{ij} + \bar{y}_{ij} \geq 1\}$; $\mathcal{A}^0 = \mathcal{A}\backslash\mathcal{A}^{pot}$; $\mathcal{A}^1 \leftarrow \emptyset$
8: for $t \in T$ do
9:     $t^{\max} \leftarrow \min\{t+1, |T|\}$
10:    $\hat{f}_{ij} \leftarrow \{w^t f^t_{ij} + \frac{1-w^t}{|T|-t^{\max}+1}\sum_{l=t^{\max}}^{|T|} f^l_{ij},\ \forall(i,j)\in\mathcal{A}^{pot};\ 0, \forall(i,j)\in\mathcal{A}^1\}$
11:    $\hat{c}^k_{ij} \leftarrow c^k_{ij}\frac{\max_t d^{kt}}{\frac{1}{|T|}\sum_t d^{kt}}$; $\hat{d}^k \leftarrow d^{kt}$
12:    $\hat{y}^t_{ij} \leftarrow \textsc{SlvSnPer}(\hat{d}^k, \hat{f}_{ij}, \hat{c}^k_{ij}, u_{ij}, b^k_i \mid y_{ij} = 1, \forall(i,j)\in\mathcal{A}^1; y_{ij} = 0, \forall(i,j)\in\mathcal{A}^0)$        ▷ Fix opened & closed arcs; solve for period $t$; store $\hat{y}^t_{ij}$
13:    for $(i,j)\in\mathcal{A}^{pot}$ do        ▷ Find open arcs
14:        if $\hat{y}^t_{ij} = 1$ then
15:            $\mathcal{A}^{pot} \leftarrow \mathcal{A}^{pot}\backslash\{(i,j)\}$        ▷ If opened, remove from potential
16:            $\mathcal{A}^1 \leftarrow \mathcal{A}^1 \cup \{(i,j)\}$        ▷ Save opening
17:        end if
18:    end for
19: end for
20: $(y^t_{ij}, x^{kt}_{ij}, v_H) \leftarrow \textsc{SlvMltPerLP}(d^{kt}, f^t_{ij}, c^k_{ij}, u_{ij}, b^k_i \mid y^t_{ij} = \hat{y}^t_{ij}, \forall(i,j)\in\mathcal{A}, \forall t\in T)$
```

abbreviated as S&T.

Selecting good arcs. We first solve two single-period instances. The first one outputs a good set of candidate arcs, by incorporating cost and demand information from the entire horizon, and the second one adopts a worst-case demand perspective, to make sure the proposed set of arcs leads to feasible solutions. We start by calculating weights w^t that are used to average the fixed cost of each arc over the horizon. The weight of each period is determined in two steps. First, we set it equal to the fraction of the remaining cumulative demand over the total demand, and then we normalize between zero and one (line). The first single-period model we solve has a fixed cost per arc (i, j) that is a weighted average of that arc's cost across the horizon, i.e., $\hat{f}_{ij} = \sum_t w^t f_{ij}^t$ (line). In order to capture future demand growth, we use the first period demand of each commodity scaled by $\max_t d^{kt} / \frac{1}{T} \sum_l d^{kl}$ (line). That way we obtain an instance with fixed costs that are more representative of the entire horizon, and demands that anticipate the growth of future periods. After solving this problem, we store which arcs are opened (line). These arcs may not be enough to guarantee feasibility throughout the horizon, and to tackle this we create an instance that has no arc opening cost and maximum demand from each commodity (line). This can be seen as a "worst-case" scenario from a demand perspective, because all commodity demands take their maximum values simultaneously. This instance can be solved efficiently as follows: since there are no arc opening costs, we solve an LP that allows positive flows via all arcs, and then select those arcs that have a strictly positive flow in the optimal solution. After solving this formulation (line), we form the set of arcs that are selected in either the first $(\hat{y}_{ij})$ or the second $(\bar{y}_{ij})$ model (line). After this initial selection of arcs, i.e., $\mathcal{A}^{pot}$, is determined, the decision of *when* to open an arc is considered.

Arc opening timing. In order to decide when to open arcs, we iteratively solve single-period instances for each period in the horizon, using only arcs from the set $\mathcal{A}^{pot}$, whose fixed cost is a weighted average of each period's cost and the average cost of the remaining periods, and the variable cost is inflated by $\max_t d^{kt} / \frac{1}{|T|} \sum_t d^{kt}$ (lines –). This way we take into account the variable cost of future periods, anticipating the potential increase in demand. Every time an arc is newly opened, we mark it as such and do not consider its cost further (lines –). During early periods, we give more weight to the actual fixed costs rather than the anticipated future costs, acknowledging that early capacity opening decisions are more important, since they impact the routing decisions of the entire horizon. Thus, fixed costs for the first periods carry the largest weights, while for later periods the weights of each period become smaller. Having decided when to open each arc, the problem reduces to solving a series of linear multi-commodity flow problems for each period in the planning horizon (line).

The advantage of the S&T heuristic is that although it works with a reduced problem size, it takes into account information from multiple periods. We thus use it to efficiently warm-start the decomposition schemes we develop.

5. Solution methods

We next exploit structural characteristics of capacitated and uncapacitated variants to utilize Lagrange relaxation and Benders decomposition, respectively.

5.1. Capacitated problems

5.1.1. Lagrange relaxation of M-NEP

By dualizing constraints (2) in the objective function (1), M-NEP decomposes into a series of single-arc multi-period subproblems. We let π_i^{kt} denote the dual values associated with constraints (2). To miminize notation clutter, we remove the indices (i, j) and formulate the single-arc problems as follows:

$$v_{ij}^{\pi} = \min \sum_{t\in T} \sum_{k\in\mathcal{K}} (c^k d^{kt} + \pi_i^{kt} - \pi_j^{kt}) x^{kt} + \sum_{t\in T} f^t y^t \qquad [SUB] \tag{8}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{K}} d^{kt} x^{kt} \le u \sum_{l=1}^{t} y^l, \qquad \forall t \in T \tag{9}$$

$$x^{kt} \le \min\{1, \frac{u}{d^{kt}}\} \sum_{l=1}^{t} y^l, \qquad \forall k \in \mathcal{K}, \forall t \in T \tag{10}$$

$$\sum_{t\in T} y^t \le 1, \tag{11}$$

$$0 \le x^{kt} \le 1, \qquad \forall k \in \mathcal{K}, t \in T \tag{12}$$

$$y^t \in \{0, 1\}, \qquad \forall t \in T. \tag{13}$$

Then, the Lagrange dual optimization problem can be expressed as

$$v_{LR}^* = \max_{\pi} v(\pi) = \max_{\pi} \left\{ \sum_{(i,j)\in\mathcal{A}} v_{ij}^{\pi} - \sum_{t\in T} \sum_{i\in\mathcal{N}} \sum_{k\in\mathcal{K}} b_i^k \pi_i^{kt} \right\}. \tag{14}$$

It is well-known that (14) is a concave optimization problem and that $v(\pi)$ is piece-wise linear (Fisher, 1981). In order to calculate v^*_{LR}, we evaluate $v(\pi)$ pointwise and apply subgradient optimization. To this end, we note the following remark.

Remark 1. For a given vector with $y^t \in \{0,1\},\ t \in T$ values that satisfy (11), problem $[SUB]$ decomposes into a series of single-period, linear bounded knapsack problems, each of which can be solved in $\mathcal{O}(|\mathcal{K}|\log|\mathcal{K}|)$ time. Given such a vector, $[SUB]$ can be solved in $\mathcal{O}(|\mathcal{K}||T|\log|\mathcal{K}|)$ time.

We next provide information on how strong the bound obtained by the Lagrange relaxation is. To this end, we note that the linear relaxation of [SUB] has the *integrality property*, i.e., its basic feasible solutions have $y^t \in \{0,1\}$ for all $t \in T$ without imposing the integrality restrictions explicitly (Fisher, 1981).

Proposition 1. *Problem [SUB] has the integrality property, thus* $v^*_{LP} = v^*_{LR}$.

Proof. See Appendix A of the electronic companion. □

5.1.2. Lower bounds and approximate primal solutions

When solving the Lagrange dual problem (14) with regular subgradient optimization, the obtained primal solution typically violates the dualized constraints (2). Our aim is to leverage the information of this solution to find promising areas of the search space and apply local-search heuristics therein. This typically involves fixing a number of variables, where the fixing decisions are driven by the values of the LP relaxation, when one is available. However, devising good quality fixing rules from the solution of (14) is challenging, because the arc opening decisions obtained by subgradient optimization (y^t_{ij}) are integral. To tackle this issue, we employ the volume algorithm of Barahona and Anbil (2000), an extension of the subgradient algorithm that returns, alongside the Lagrangian lower bound, a y-solution with small violations of (2). This way, we have a solution with fractional binary variables at each subgradient iteration, which we then use to direct our heuristic search.

5.1.3. Finding upper bounds

Within the framework of the volume algorithm we find feasible solutions of the original problem, M-NEP, by employing three local search heuristics. The first heuristic checks if moving the arc opening decisions later in time provides an improved solution. Algorithm 2 shows the details.

Algorithm 2 Single-arc search heuristic.

```
Input: Period $t$, feasible solution $(\bar{y}^t_{ij}, \bar{x}^{kt}_{ij})$, objective value $z^t$
Output: Improved feasible solution $(\bar{y}^t_{ij}, \bar{x}^{kt}_{ij})$, objective value $z^t$
 1: $\bar{\mathcal{A}} \leftarrow \{(i,j) \in \mathcal{A} : \bar{y}^t_{ij} = 1\}$
 2: $M^t \leftarrow$ LP model for period $t$ with $x^{kt}_{ij} \le \sum_{l=1}^{t} \bar{y}^l_{ij}, \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}$
 3: for $(i,j) \in \bar{\mathcal{A}}$ do
 4:     $\Delta fc \leftarrow f^t_{ij} - f^{t+1}_{ij}$
 5:     Solve $M^t$ with $x^{kt}_{ij} = 0, \forall k \in \mathcal{K}$
 6:     $z' \leftarrow$ Objective$(M^t)$ if $M^t$ is feasible, otherwise $z' = \infty$; $\Delta z \leftarrow z' - z^t$
 7:     If $\Delta fc - \Delta z > 0$ then                ▷ Fixed cost reduction > flow cost increase
 8:         $\bar{y}^t_{ij} \leftarrow 0, \bar{y}^{t+1}_{ij} \leftarrow 1, \bar{x}^{kt}_{ij} \leftarrow$ Solution$(M^t), z^t \leftarrow \sum_{(i,j)\in\bar{\mathcal{A}}} c^k_{ij} d^{kt} \bar{x}^{kt}_{ij}$
 9: end for
10: return $(\bar{y}^t_{ij}, \bar{x}^{kt}_{ij}, z^t)$
```

For large models, however, attempting to check all arcs might be prohibitive. We thus restrict the search to a neighborhood defined by comparing the arc-opening variables of the best feasible solution (the incumbent) and the possibly fractional arc-opening variables, retrieved by the volume algorithm. Concretely, for a cutoff point $\delta \in (0,1)$ and for each arc (i,j), we set $y^{t^*}_{ij} = 1$ if $t^* = \text{argmax}_t\{y^t_{ij} : y^t_{ij} > \delta\}$ exists and $y^t_{ij} = 0, \forall t \in T$, otherwise. Then, we compare the resulting vector with the incumbent solution. For each period, we invoke Algorithm 2 only for the arcs that are different in the incumbent and the rounded solution. In addition, we sort the arcs in order of decreasing Δfc, in order to ensure that we first check those that give the highest period-on-period reduction in fixed costs. This procedure utilizes a small set of relevant arcs, for which there are discrepancies between the rounded approximate primal solution of the Lagrange relaxation and the incumbent solution.

Our second heuristic checks if changing the period when an arc is opened leads to improved solutions. Thus, given a feasible solution $\bar{y}^t_{ij}$ we define $\bar{\mathcal{A}}_t = \{(i,j) \in \mathcal{A} | \bar{y}^t_{ij} = 1\}, \forall t \in T$ and impose the constraints

$$\sum_{l=\max\{t-\tau^-,1\}}^{\min\{|T|,t+\tau^+\}} y^l_{ij} = 1, \quad \forall t \in T, \forall (i,j) \in \bar{\mathcal{A}}_t \tag{15}$$

to the original problem M-NEP. This explores a neighborhood of the incumbent where the arc opening decisions remain the same, but their timing can be shifted up to τ^+ periods later or τ^- periods earlier. Note that this heuristic is a local branching procedure (Fischetti and Lodi, 2003) in which we impose a customized search neighborhood structure.

Our third and last heuristic applies a simple fixing procedure based on the values of the fractional y variables recovered by the volume algorithm, as follows. First, it interprets each fractional value as signifying a degree of "ambiguity", meaning that the exploration of variables closer to 0.5 takes priority. To this end, it sorts the y variables in increasing values of $|y^t_{ij} - 0.5|$ and, for a given fraction f, it finds l_f and u_f in (0,1) such that the first $f|\mathcal{A}||T|$ variables lie in the interval $[l_f, u_f]$. Finally, in M-NEP, it fixes to zero those variables that are lower than l_f and to one those that are higher than u_f, respectively, and solves the remaining MIP model. The advantage of this heuristic is that it can be tuned easily by only changing f, while it searches a promising neighborhood of the fractional solution.

5.2. Uncapacitated problems

Removing constraints (3) from M-NEP gives rise to the uncapacitated variant, M-UNEP. The resulting model exhibits a decomposable structure: for fixed arc opening decisions, it decomposes into a series of independent shortest path problems, per commodity and per period. In this part, we leverage this property to develop Benders decomposition formulations.

5.2.1. Regular benders decomposition.

Let Y denote the set of binary vectors y that satisfy $\sum_{t\in T} y^t_{ij} \le 1, \forall (i,j) \in \mathcal{A}$. Then, for a given $\bar{y} \in Y$, the uncapacitated model reduces to the following linear program:

$$z(\bar{y}) = \min \sum_{t\in T}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}} c^k_{ij} d^{kt} x^{kt}_{ij} \tag{16}$$

$$\text{s.t} \sum_{j:(i,j)\in\mathcal{A}} x^{kt}_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x^{kt}_{ji} = b^k_i, \qquad [\pi^{kt}_i],\ \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in T, \tag{17}$$

$$0 \le x^{kt}_{ij} \le \sum_{l=1}^{t} \bar{y}^l_{ij}, \qquad [\lambda^{kt}_{ij}],\ \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in T, \tag{18}$$

Then, the original problem can be expressed as $\min_{y\in Y}\{z(y) + \sum_{t\in T}\sum_{(i,j)\in\mathcal{A}} f^t_{ij} y^t_{ij}\}$. Note that (16)–(18) decomposes into a series of shortest path problems, each one corresponding to a commodity–period pair. The dual of (16)–(18) is then

$$z(\bar{y}) = \max \sum_{t\in T}\sum_{k\in\mathcal{K}} \left(\pi^{kt}_O - \pi^{kt}_D\right) - \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}}\sum_{t\in T} \Big(\sum_{l=1}^{t} \bar{y}^l_{ij}\Big)\lambda^{kt}_{ij} \tag{19}$$

$$\text{s.t. } \pi^{kt}_i - \pi^{kt}_j - \lambda^{kt}_{ij} \le c^k_{ij} d^{kt}, \qquad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, t \in T \tag{20}$$

$$\lambda^{kt}_{ij} \ge 0, \qquad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, t \in T. \tag{21}$$

The dual polyhedron $\Delta = \{(\boldsymbol{\pi}, \boldsymbol{\lambda}) | (20) - (21)\}$ is non-empty, and therefore can be represented by a finite set of extreme rays, denoted by R_Δ, and a finite set of extreme points, denoted by P_Δ (Schrijver, 1998). Using this representation, the Benders reformulation of M-UNEP is as follows:

$$\min \sum_{t\in T}\sum_{(i,j)\in\mathcal{A}} f^t_{ij} y^t_{ij} + z \tag{22}$$

$$\text{s.t.} \sum_{t\in T}\sum_{k\in\mathcal{K}} \left(\bar{\pi}^{kt}_O - \bar{\pi}^{kt}_D\right) - \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}}\sum_{t\in T} \left(\sum_{l=t}^{|T|} \bar{\lambda}^{kl}_{ij}\right) y^t_{ij} \le 0, \qquad \forall (\bar{\pi}^{kt}_i, \bar{\lambda}^{kt}_{ij}) \in R_\Delta \tag{23}$$

$$\sum_{t\in T}\sum_{k\in\mathcal{K}} \left(\bar{\pi}^{kt}_O - \bar{\pi}^{kt}_D\right) - \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}}\sum_{t\in T} \left(\sum_{l=t}^{|T|} \bar{\lambda}^{kl}_{ij}\right) y^t_{ij} \le z \qquad \forall (\bar{\pi}^{kt}_i, \bar{\lambda}^{kt}_{ij}) \in P_\Delta \tag{24}$$

$$\sum_{t\in T} y^t_{ij} \le 1, \qquad \forall (i,j) \in \mathcal{A} \tag{25}$$

$$y^{kt}_{ij} \in \{0,1\}, \qquad \forall (i,j) \in \mathcal{A}, \forall t \in T. \tag{26}$$

Constraints (23), the *feasibility cuts*, prevent the objective function of the dual problem (19)–(21) from being unbounded, and therefore the corresponding primal problem (16)–(18) from becoming infeasible. Constraints (24) are the *optimality cuts*, which impose that z corresponds to the optimal objective value function of the dual subproblem (19)–(21). This formulation is conceptually useful, but since the cardinalities of P_Δ and R_Δ are usually large, adding all constraints (23) and (24) is computationally inefficient. Benders himself noted that (22)-(26) can be solved for a limited number of feasibility and optimality cuts, in which case it delivers a lower bound on the optimal objective function value, and new cuts can be added dynamically. Specifically, in each iteration the optimal y^t_{ij} values can be used to solve the pair of primal-dual subproblems (16)–(18) and (19)–(21), respectively. If the primal subproblem is infeasible, then the dual subproblem returns a feasibility cut, (23), whereas when the primal subproblem is feasible, the dual subproblem returns an optimality cut, (24). These cuts are added to the master problem and the algorithm proceeds to the next iteration. If no optimality or feasibility cut is violated, then the algorithm has converged to an optimal solution.

Modern implementations of Benders decomposition employ additional computational enhancements. First, the decomposition of the subproblem into a series of $|\mathcal{K}||T|$ subproblems allows the generation of individual cuts from each commodity–period pair. To this end, z is replaced by $\sum_{k\in\mathcal{K}}\sum_{t\in T} z^{kt}$ in (22), the summations over periods and commodities are dropped in (23) and (24), and the sets of extreme rays and points are defined over the polyhedra $\Delta^{kt} = \{(\boldsymbol{\pi},\boldsymbol{\lambda}) \mid \pi^{kt}_i - \pi^{kt}_j - \lambda^{kt}_{ij} \le c^k_{ij} d^{kt};\ \lambda^{kt}_{ij} \ge 0, \forall (i,j)\in\mathcal{A}\}$. The corresponding cuts are denser and tend to be more effective than a single cut generated from the aggregated master problem (22)–(26) (Cordeau et al., 2001). Second, instead of solving the master program (22)–(26) to optimality in each iteration, we take advantage of the callback capabilities of modern solvers to solve it only once and add the cuts dynamically in the branch-and-bound tree. Concretely, we start by solving the master program with a limited number of cuts, use a callback to invoke the cut-generating procedure every time a feasible solution is found and to add the generated cuts, and then return control to the solver (Bai and Rubin, 2009; Adulyasak et al., 2015). In addition, we add cuts from individual subproblems, and two classes of valid inequalities which warm-start the master problem, as detailed in Section 5.2.5.

5.2.2. FSZ Benders decomposition

Fischetti et al. (2010) suggest an alternative normalization approach that uses the best known flow cost values $\bar{z}^{kt}$ in the subproblem formulation. Let $(\bar{y}^t_{ij}; \bar{z}^{kt})$ denote a feasible solution of the master problem (22)–(26) with disaggregated Benders cuts. For notational brevity, we suppress the commodity and period notation when we refer to a single subproblem. Using this notation, Fischetti et al.'s subproblem (FSZ) can be formulated as follows:

$$\max \quad \pi_O - \pi_D - \sum_{(i,j)\in\mathcal{A}} \Big(\sum_{l=1}^{t} \bar{y}^l_{ij}\Big)\lambda_{ij} - \bar{z}\eta \qquad [FSZ] \tag{27}$$

$$\text{s.t.} \quad \pi_i - \pi_j - \lambda_{ij} \le c_{ij} d\eta, \qquad \forall (i,j)\in\mathcal{A} \tag{28}$$

$$\sum_{(i,j)\in\mathcal{A}} w_{ij}\lambda_{ij} + w_0\eta = 1 \tag{29}$$

$$\eta \ge 0;\ \ \lambda_{ij} \ge 0, \qquad \forall (i,j)\in\mathcal{A}. \tag{30}$$

In this formulation, the user is able to select non-negative weights w_{ij} and w_0 to better configure the normalization hyperplane (29). The regular Benders subproblem arises as the special case of $w_0 = 1$ and $w_{ij} = 0$. Our implementation sets the convexity condition $w_0 = w_{ij} = 1$ for each arc $(i,j)\in\mathcal{A}$. As long as non-negative weights are selected, FSZ always has a feasible solution and is bounded. Therefore, a cut $\pi_O - \pi_D - \sum_{(i,j)\in\mathcal{A}}(\sum_{l=1}^{t}\bar{y}^l_{ij})\lambda_{ij} - \bar{z}\eta \le 0$ can always be generated. Note also that if no arc has been opened until period t, i.e., $\sum_{l=1}^{t}\bar{y}^l_{ij} = 0$ for each $(i,j)\in\mathcal{A}$, an optimal solution will generate a cut with $\eta = 0$ and $\sum_{(i,j)\in\mathcal{A}}\lambda_{ij} = 1$. Thus, this subproblem can generate both optimality and feasibility cuts.

5.2.3. Pareto-optimal cuts

The cut selection problem becomes relevant when the dual subproblem (19)–(21) has multiple optimal solutions, and therefore one has to select the best among alternative cuts. A criterion that partially quantifies cut quality is *cut dominance* (Magnanti and Wong, 1981): a cut generated from the extreme point $(\boldsymbol{\pi}^1,\boldsymbol{\lambda}^1)$ is said to dominate another cut generated from $(\boldsymbol{\pi}^2,\boldsymbol{\lambda}^2)$ iff

$$\pi^1_O - \pi^1_D - \sum_{(i,j)\in\mathcal{A}}\left(\sum_{l=1}^{t} y^l_{ij}\right)\lambda^1_{ij} \ge \pi^2_O - \pi^2_D - \sum_{(i,j)\in\mathcal{A}}\left(\sum_{l=1}^{t} y^l_{ij}\right)\lambda^2_{ij}$$

holds for all $y^t_{ij} \in Y = \{y^t_{ij} \in \{0,1\} \mid \sum_{t\in T} y^t_{ij} \le 1, \forall (i,j)\in\mathcal{A}\}$ with strict inequality for at least one point. A cut is non-dominated, or *Pareto optimal* (PO) if there is no other cut that dominates it. Magnanti and Wong (1981) devised a mechanism that generates PO cuts by solving an additional linear program, formulated as follows. First, let $\boldsymbol{y}^r = (y^{t,r}_{ij})^{t\in T}_{(i,j)\in\mathcal{A}}$ denote a *core point*, i.e., a point that lies in the relative interior of $conv(Y)$. Then, denoting by $\bar{y}^t_{ij}$ the master problem solution and by $z^* = \max\{\pi_O - \pi_D - \sum_{(i,j)\in\mathcal{A}} \sum_{l=1}^t \bar{y}^l_{ij}\lambda^t_{ij} : (\boldsymbol{\pi}, \boldsymbol{\lambda}) \in \Delta^{kt}\}$ the subproblem solution for the pair (k, t), one can identify a PO cut by solving the following subproblem:

$$\max \quad \pi_O - \pi_D - \sum_{(i,j)\in\mathcal{A}} \left(\sum_{l=1}^t y^{l,r}_{ij}\right)\lambda_{ij} \qquad [PO-SUB] \tag{31}$$

$$\text{s.t.} \quad \pi_i - \pi_j - \lambda_{ij} \le c_{ij}d, \qquad \forall (i,j)\in\mathcal{A} \tag{32}$$

$$\pi_O - \pi_D - \sum_{(i,j)\in\mathcal{A}} \left(\sum_{l=1}^t \bar{y}^l_{ij}\right)\lambda_{ij} = z^* \tag{33}$$

$$\lambda_{ij} \ge 0 \qquad \forall (i,j)\in\mathcal{A}. \tag{34}$$

Constraint (33) ensures that the new point is an optimal solution to the original subproblem, while the objective function ensures that it is a PO cut.

Although finding a core point is $\mathcal{NP}$-hard in general (Papadakos, 2008), the simple structure of Y makes it possible to characterize a family of core points: given a feasible point, $\bar{y}^t_{ij} \in Y$, the perturbed point $y^{t,r}_{ij} = \{1 - \epsilon|T| \text{ if } \bar{y}^t_{ij} = 1; \epsilon, \text{ otherwise}\}$ is a core point for $\epsilon \in (0, 1/|T|)$. Selecting a small ϵ guarantees that we generate a core point which lies in the neighborhood of $\bar{y}^t_{ij}$. Thus, we make use of this selection policy in our implementation. We also note that an optimal solution to the regular Benders subproblem is feasible to the PO subproblem.

5.2.4. Efficient generation of Pareto-optimal cuts

For single-period uncapacitated problems, Magnanti et al. (1986) have shown that a PO cut can be generated by solving a single minimum cost flow problem instead of two generally structured LPs. We show here that their main argument can be extended to our setting. To this end, we rewrite the dual of (31)–(34) as follows:

$$\max \quad d\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} - z^*x_0 \tag{35}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{A}_i^+} x_{ij} - \sum_{j\in\mathcal{A}_i^-} x_{ji} = b_i(1+x_0), \qquad \forall i\in\mathcal{N} \qquad [\pi_i] \tag{36}$$

$$0 \le x_{ij} \le x_0\sum_{l=1}^t \bar{y}^l_{ij} + \sum_{l=1}^t y^{l,r}_{ij}, \qquad \forall (i,j)\in\mathcal{A}\ [\lambda_{ij}\ge 0], \tag{37}$$

where x_0 represents the dual price of constraint (33). Magnanti et al. (1986) observe that x_0 can be fixed to any value greater than or equal to $\sum_{(i,j)\in\mathcal{A}}\sum_{l=1}^t y^{l,r}_{ij}$ at an optimal solution. In our computational experiments, we set $x_0 = |\mathcal{A}|$. Therefore, the objective term z^*x_0 becomes a constant and (35)–(37) is recast as a minimum cost flow problem, while its dual solution corresponds to a PO cut.

5.2.5. Strengthening the master problem

We use two families of cutting planes that strengthen the formulation of the master problem. First, we employ origin-destination cuts, which impose that at least one arc from each origin and to each destination should be opened, respectively. To explain our second set of inequalities, let p^* denote the shortest path cost for a certain commodity-period pair and p^*_{ij} the shortest path cost when arc (i, j) is removed from the graph. Then, the cut $z \ge p^*_{ij} + (p^* - p^*_{ij})\sum_{l=1}^t y^l_{ij}$ is a valid cut (Magnanti et al., 1986), since it expresses the fact that the flow cost for a commodity cannot be lower than the shortest path cost when the complete graph is considered ($\sum_{l=1}^t y^l_{ij} = 1$) or p^*_{ij} when arc (i, j) is not opened ($\sum_{l=1}^t y^l_{ij} = 0$). In our implementation, instead of adding $|\mathcal{T}||\mathcal{K}||\mathcal{A}|$ such inequalities, we find for each commodity the arc whose removal increases the shortest path cost the most and add inequalities only for this arc.

6. Computational experiments

The purpose of our computational study is twofold. First, we aim to illustrate the efficiency of the developed solution methods. To this end, we conduct a series of experiments that assess the quality of (i) the lower bound obtained by Lagrange relaxation (LR); (ii) the upper bound returned by the S&T heuristic; (iii) the LR algorithm integrated with S&T and the heuristics described in Section 5.1.3 and (iv) the various Benders implementations. Second, we investigate how different instance characteristics influence the solution structure. In particular, we are interested in deriving insights on *when* arcs are opened, what is the influence of the fixed versus variable cost ratio, how capacity tightness influences the timing of arc opening and how cost correlations influence solution characteristics.

Table 2
Characteristics of the networks of Pazour et al. (2010). The variable range refers to the generated multi-period instances, for $|T| = 5$ and $|T| = 20$, respectively.

Instance	$\vert\mathcal{N}\vert$	$\vert\mathcal{A}\vert$	$\vert\mathcal{K}\vert$	Variable range	Constraint range
USC30	30	126	87	[55,440 - 221,760]	[13,680 - 54,720]
USC53	53	278	245	[341,940 - 1,367,760]	[66,315 - 265,260]
JBH50	50	198	62	[620,730 - 2,482,920]	[157,490 - 629,960]

Table 3
Design parameters of single-period network design instances.

Parameter	Levels	Symbols	Explanations
Fixed cost	Low, Medium, High	L, M, H	Ratio $fr = \frac{\sum f_{ij}}{\sum d^k \sum c_{ij}} \in \{0.01, 0.5, 0.1\}$, respectively
Capacity	Loose, Medium, Tight	L, M, T	Ratio $cr = \frac{\vert\mathcal{A}\vert \sum_{k\in\mathcal{K}} d^k}{\sum u_{ij}} \in \{1, 2, 8\}$, respectively
Correlation structure	Positive correlation (Original), Negative correlation	PC, NC	If negative, fixed and variable costs have correlation -70%
Routing cost mode	Euclidean, Mixed, Random	E, M, R	Costs proportional to distance (E), 50% of costs shuffled randomly (M), all costs shuffled randomly (R)

6.1. Instances

We utilize three real-world networks introduced in Pazour et al. (2010), which originate from high-speed rail network design for cargo distribution. Specifically, two of these networks are constructed using data from the US Census Bureau and one from the annual shipments of J.B. Hunt transport services. We use these networks to construct 648 instances with horizons varying from 5 to 20 periods, 624 of which are feasible for capacitated problems. Pazour et al. (2010) use a single-period network design formulation but recognize that "*due to the high costs of these systems, it is likely that a high-speed network, [...], would be implemented in phases throughout a planning horizon of many years*". Therefore, these instances are appropriate use cases for our formulation. We then repeat our analysis using a subset of the R instances constructed by Crainic et al. (2001), which we extend to 20, 40, 60 and 80 periods, for a total of 432 instances, 408 of which are feasible for capacitated problems. Although such long horizons are rarely found in transportation networks, they are relevant in problems arising in telecommunication networks, such as in Idzikowski et al. (2011), where the authors use 15-minute epochs to analyze networks with daily dynamic demand, resulting in problems with 96 periods. For brevity, we have included the detailed analysis of this experiment in the electronic companion, and give a summary of results in the main paper, focusing on the instances originating from real transportation networks. In total, our complete dataset consists of 1080 instances, which are used to assess algorithmic performance and gain insights in the problem structure. Next, we provide a brief overview of the multi-period Pazour networks. Further details of how the instances are constructed and to what cost structure each label corresponds can be found in Appendix B.1 of the electronic companion of this paper.

In order to keep the instances tractable in a multi-period setting, we have kept commodities that cover 80% of the total original demand by eliminating the commodities with the smallest demand. Table 2 shows the characteristics of these instances. Gurobi is able to eliminate about 4% of variables via pre-processing, for capacitated instances using the M-CNDP formulation. Therefore, the reported model sizes are accurate representations of the actual model sizes tackled by Gurobi.

We used these three instances and the methodology in Crainic et al. (2001) to construct instances with loose, medium and tight capacities and with low, medium and high fixed cost ratios. In addition, we further extend the design space by considering instances that have correlations between fixed and variable costs and instances where these costs can be proportional to the original distance matrix, random or mixed. Table 3 presents in detail the levels of each parameter we considered.

Using a full factorial design, we have constructed 54 instances from each original instance, which we then extended to multiple periods. The fixed cost per arc is assumed to decrease linearly with the remaining periods, such that (i) the average fixed cost per arc equals that of the single-period instance, and (ii) the last period has 10% of the single-period fixed cost. For demand expansion, we use a sigmoid curve which consists of a convex, a linear and a concave part. This generic profile represents a period of rapid growth in demand, a subsequent linear trend and then a stabilization phase. The curves are constructed so that the average demand coincides with that of the original instance, and that demand expands from 50% to 150% of the original demand. Further details can be found in the electronic companion of this paper.

Table 4
Average Lagrange relaxation gaps and CPU Time for instances where the exact lower bound is found (columns 1–6) and where Gurobi returns another lower bound at the root node (columns 7–10).

(1) Instances	(2)	(3) GRB	(4) LR<GRB LP	(5) CPU Time (s)	(6)	(7) LR>GRB Root	(8) LR<GRB Root	(9) CPU Time (s)	(10)
$\lvert\mathcal{N}\rvert$	#	Solved	LB Gap (%)	GRB	LR	LB Gap (%)	LB Gap (%)	GRB	LR
30	216	216	0.43	**4**	147	0.09 (59)	0.61 (157)	**33**	147
50	164	97	0.86	1301	**733**	16.11 (56)	1.17 (108)	3652	**913**
53	216	181	2.16	991	**639**	9.68 (48)	2.20 (168)	3473	**683**
$\lvert T\rvert$									
5	159	151	0.23	670	**278**	2.59 (29)	0.27 (130)	1828	**289**
10	155	130	0.76	670	**422**	6.40 (52)	0.89 (103)	2306	**478**
15	148	119	1.69	638	**578**	12.54 (41)	1.80 (107)	2480	**681**
20	134	94	2.48	**447**	564	10.98 (41)	2.94 (93)	1828	**808**
Total	596	494	1.15	620	**442**	8.42 (163)	1.37 (433)	2276	**552**
Total (ℝ)	408	257	0.85	760	**166**	12.83 (101)	1.10 (283)	4258	**206**

Notes. Columns (4) and (8) are calculated as $(LB_{GRB} - LB_{LR})/LB_{GRB}$, while column (7) as $(LB_{LR} - LB_{GRB})/LB_{LR}$. Columns (9) and (10) report the average CPU time for instances in (7) and (8) combined. Numbers in parentheses denote the number of instances. For the Pazour instances, we report results for 596 out of 624 instances, because Gurobi ran out of memory or returned a segmentation fault at the remaining 28 instances.

6.2. Computational performance

The experiments of this section aim to illustrate the usefulness of the developed algorithms, and to benchmark them against solving the MIP formulation (1)–(7). When it comes to the inclusion of the strong inequalities (4) in the LP relaxation of (1)–(7), we implemented and tested four alternative strategies: (i) not including strong inequalities; (ii) adding all of them a priori in the model; (iii) adding them dynamically (via a callback) when they are violated, and (iv) adding them in a lazy cut pool and resorting to Gurobi's pool management mechanism. Note that Gurobi does not allow to use lazy cuts for an LP. Therefore, we formulated the problem as a MIP and added the constraints as lazy constraints. Then, Gurobi thinks we solve a MIP model but in practice we solve the LP relaxation, by setting the parameter 'cuts' to zero and setting the node limit to one. Strategy (iv) gave the best results overall, when considering CPU time and optimality gap quality, and therefore we adopt this strategy throughout our computational experiments. To this end, we study capacitated and uncapacitated variants separately, since they utilize different solution methods. In each table, we report results disaggregated per number of nodes and periods for the Pazour instances, and provide aggregate results for the ℝ instances.

6.2.1. Capacitated instances

The first matter of interest is to assess the quality of the lower bound returned by Lagrange relaxation (LR). In theory, the Lagrange dual provides the same lower bound as the LP relaxation of (1)–(7) when the tight inequalities (4) are included as lazy cuts. In practice, when LR is solved by subgradient optimization a lower approximation is obtained, which may or may not be close to the LP bound, depending on implementation-specific details (Fisher, 1981). We therefore solve the LP relaxation of (1)-(7) with Gurobi, adding all strong inequalities as default constraints, and compare this bound with the one obtained by LR. Table 4 reports the results, showing in column (3) the number of instances for which Gurobi could solve the LP relaxation within a time limit of 7200s, in column (4) the lower bound gap, defined as $(LB_{GRB} - LB_{LR})/LB_{GRB}$, and the CPU time consumed by Gurobi and LR in columns (5) and (6), respectively, for the instances solved to optimality. These results suggest that the lower bound returned by LR is close to the exact bound, having a gap of approximately 1%. For the Pazour instances, Gurobi consumes 40% more CPU time, and although it is faster than LR for small instances, its performance does not scale up equally well for larger instances. To see this, note that for 20-period instances Gurobi fails to solve the LP relaxation for 40 out of 134 instances, while for the remaining 94 instances it is marginally faster than LR. Furthermore, for the ℝ instances with 20 to 80 periods, the CPU time of LR is one fifth of the CPU time for Gurobi.

Next, we compare the LR bound against the actual bound Gurobi returns at the root node, in columns (7) to (10) of Table 4. The reason for doing so is that Gurobi does not necessarily utilize all strong inequalities when solving the root node of the MIP formulation, but rather a subset of them combined with generic cutting planes. In fact, while the LP relaxation with strong inequalities could not be solved within the time limit for 130 Pazour instances and for 151 ℝ instances, when solved as a MIP Gurobi was able to return a lower bound at the root node for 596 Pazour instances and 384 ℝ instances. However, since not all strong inequalities are included by default, this lower bound is no longer guaranteed to be better than the one of LR. This leads us to partition the instances in those where LR or GRB returned the best lower bound, respectively, and assess each category separately.

Column (7) reveals that in cases where LR returns a stronger lower bound for the Pazour instances, this is 8.42% stronger than that of Gurobi, while column 8 suggests that when Gurobi finds a stronger lower bound this is only 1.37% stronger than that of LR. These conclusions also apply to the ℝ instances. Overall, Gurobi is four times slower for the Pazour and 20 times slower for the ℝ instances, respectively, in calculating a root node lower bound compared to LR. These findings are qualitatively persistent across planning horizons of various lengths. Although Gurobi seems more efficient for the USC30 in-

Table 5
Upper bound quality of various heuristics. Gaps are calculated as $(UB_{heur} - UB_{GRB})/UB_{GRB}$. The last column shows total time and, in parenthesis, the time Gurobi consumed to find the optimal solution.

		UB Gap (%)				CPU Time (s)				
$\lvert\mathcal{N}\rvert$	Optimal	MH	RH(5,1)	S&T	LR	MH	RH(5,1)	S&T	LR	GRB
30	216	14.37	1.30	4.09	**0.39**	12	107	**9**	189	128 (81)
53	117	12.98	2.75	4.80	**0.47**	767	2077	**209**	1277	1389 (1171)
50	89	6.59	3.78	2.99	**0.16**	854	1749	**300**	1599	1415 (1312)
$\lvert T\rvert$										
5	110	5.81	1.57	9.41	**0.35**	312	632	**92**	495	500 (439)
10	111	11.7	2.34	3.17	**0.36**	410	890	**142**	752	559 (598)
15	107	14.79	2.39	1.82	**0.36**	493	1231	**162**	995	716 (849)
20	94	17.96	2.67	1.38	**0.41**	381	1296	**103**	938	832 (699)
Total	422	12.34	2.23	4.06	**0.37**	399	999	**125**	788	649 (643)
Total (ℝ)	163	27.88	2.85	4.84	**0.86**	249	1634	**47**	332	2092 (1591)

Table 6
Optimality gap quality of LR. Gaps are calculated as $(UB - LB)/UB$. The last two columns refer to instances in which Gurobi terminated after 7200 s without an upper bound.

	Instances		Integrality Gap (%)		CPU Time (s)		Only LR	
$\lvert\mathcal{N}\rvert$	All	GRB Gap<1	GRB	LR	GRB	LR	Gap (%)	CPU Time (s)
30	216	216	**0.00**	0.96	**128**	189	-	-
50	192	128	4.61	**2.11**	1953	**1726**	2.63	2849
53	216	192	5.49	**4.26**	2481	**1608**	6.88	2440
$\lvert T\rvert$								
5	159	153	3.13	**1.80**	2121	**825**	7.48	2033
10	156	141	3.18	**1.87**	1966	**1048**	2.70	2838
15	155	129	3.03	**2.71**	1836	**1235**	4.09	2949
20	154	113	**2.89**	3.58	1854	**1215**	4.75	2532
Total	624	536	3.07	**2.41**	1951	**1065**	4.33	2686
Total (ℝ)	408	357	7.51	**5.14**	4868	**517**	9.76	3738

stances, which are the smallest ones in size and can all be solved optimally, its advantage is not retained for larger instances. The conclusion from this experiment is that LR delivers high quality lower bounds, which are close to the theoretically best ones, while its performance scales well to large instances.

Next, we assess the quality of upper bounds, as obtained by the S&T heuristic and by the integrated LR algorithm, i.e., the LR bounding scheme combined with our incremental and local branching heuristics, when it is initialized by the S&T heuristic, as described in Section 5.1.3. To this end, we utilize the instances that Gurobi could solve to proven optimality, and compare their bounds to the ones obtained by our heuristics. In addition, we utilize two other benchmark approaches: one that myopically solves single-period problems, fixes the arcs to be opened and proceeds to the next period, and a rolling horizon one which solves a problem of length $l < T$, fixes the arc opening decisions of the first m periods and repeats, starting from period $m + 1$, until the end of the horizon. After some preliminary tuning, we select $l = 5$ and $m = 1$ when $T > 5$ and $l = 3$, $m = 1$ for $T = 5$. Table 5 reports the results.

A first observation is that the myopic heuristic (MH) delivers solutions that deviate considerably from optimality. In particular, solution quality deteriorates consistently for problems with longer horizons. The rolling horizon heuristic (RH) finds much better solutions, but also consumes a considerably larger CPU time, while its performance also deteriorates with longer horizons. The S&T heuristic finds solutions with a larger gap compared to RH, but it does so in a fraction of the time RH requires to terminate. Given that the purpose of S&T is to find good solutions to inject in LR, its performance is aligned with its objective. In addition, it delivers lower gaps for longer horizons and the main difference with RH is in five-period instances. Finally, we note that LR finds solutions of excellent quality, having an average gap of 0.37% for the Pazour and 0.86% for the ℝ instances, respectively. In terms of CPU time, LR is slightly slower than Gurobi for the small Pazour instances but six times faster for the ℝ instances. The slower performance on the Pazour instances can be partly attributed to the fact that our algorithm is tuned to be efficient across a wider set of instances. Specifically, we made some critical design choices, such as the number of subgradient iterations, so that not only a good upper bound is found but also a strong lower bound is returned consistently across all instances, at the expense of longer CPU times for easier instances.

Finally, Table 6 reports the overall performance of the LR algorithm on all instances.

Specifically, column 3 reports the number of instances in which Gurobi could find non-trivial upper and lower bounds, i.e., which have a gap less than 100%, and columns 4 and 5 the corresponding optimality gaps for each method, with respect to their own upper and lower bound. Next, columns 6 and 7 report the corresponding CPU times. The results suggest that LR delivers a better overall gap, while it consumes less CPU time compared to Gurobi. This is true for all but the smaller instances, USC30, and for instances with 20 periods. For the latter, Gurobi returns a smaller gap, but note that it does not

Table 7
Optimality gaps and CPU times of Gurobi and Benders implementations.

		Optimality Gap (%)					CPU Time (s)				
$\lvert\mathcal{N}\rvert$	#	GRB	B-REG	B-MW1	B-MW2	B-FSZ	GRB	B-REG	B-MW1	B-MW2	B-FSZ
30	216	**0.00**	0.03	0.01	0.48	**0.00**	**28**	622	777	86	**28**
50	216	48.92	23.40	9.12	3.30	**0.67**	3264	5161	5175	4171	**2240**
53	216	17.5	11.57	5.92	3.57	**1.38**	3852	6650	6818	4809	**2888**
$\lvert T\rvert$											
5	162	8.66	7.21	3.38	2.34	**1.21**	2384	3952	3910	3103	**1959**
10	162	15.43	9.62	3.54	2.57	**0.68**	2456	4107	4624	2983	**1710**
15	162	30.79	13.65	2.86	2.23	**0.49**	2334	4276	4975	3047	**1465**
20	162	33.69	16.20	10.27	2.65	**0.35**	2351	4243	3517	2956	**1630**
Total	648	22.14	11.67	5.01	2.45	**0.68**	2381	4144	4257	3022	**1696**
Total ($\mathbb{R}$)	432	15.12	6.47	2.33	1.52	**0.42**	1902	2191	2866	1583	**1572**

manage to return a gap for 41 instances. In both cases, LR remains competitive. It is worth noting that columns 3 to 7 subsume instances that Gurobi solved to optimality. If we consider instances which Gurobi could not solve to optimality but for which it did find an upper and lower bound, then the difference is even more profound: for example, for the Pazour instances LR returns an average gap of 5.09% and 2088 s of CPU time, as opposed to a gap of 14.41% and 7200 s of CPU, reported by Gurobi. Finally, the last two columns report the optimality gap and CPU time of LR when run on instances for which Gurobi could not find an upper bound. For these difficult instances, our algorithm seems to scale very well, with an average CPU time of 2686 s (3738 for the $\mathbb{R}$ instances) and an optimality gap of 4.33% (9.76% for the $\mathbb{R}$ instances). Interestingly, it seems that instances with five periods are quite challenging to solve.

In summary, these experiments suggest that the S&T heuristic and the Lagrange relaxation constitute efficient bounding techniques that are competitive with Gurobi for small instances, scale better for medium instances and attain good optimality gaps for the most difficult instances. We then proceed to uncapacitated instances and assess the performance of the various Benders decomposition implementations.

6.2.2. Uncapacitated instances

We investigate the efficiency of our Benders decomposition implementations by utilizing the same set of multi-period instances but without considering their capacity. Specifically, we benchmark (i) a modern implementation of the regular formulation (22)–(26), (B-Reg); (ii) Adding PO cuts (B-MW1); (iii) Adding PO cuts solving only one subproblem, as in Magnanti et al. (1986) (B-MW2) and finally (iv) the FSZ formulation of Fischetti et al. (2010) (B-FSZ). All implementations exploit modern callback technology to avoid solving the master problem multiple times (Bai and Rubin, 2009) and make use of the cutting planes described in Section 5.2.5, while the Pareto-optimal formulations update the core point using Remark 3. In addition, all methods other than B-FSZ use a subroutine that detects if a primal subproblem is infeasible, by either finding a path between each $o-d$ pair or by detecting an edge cut set, which is then used to generate a feasibility cut. This is not necessary for B-FSZ, since the FSZ subproblem generates both optimality and feasibility cuts. Similar to the capacitated experiments, our basic benchmark is the best Gurobi (GRB) formulation, which uses the cut pool to handle the separation of the strong inequalities (18). Table 5 shows the average optimality gaps and CPU times obtained by each method. The time limit is set to 7200 s for all algorithms.

A careful analysis of Table 7 suggests some important conclusions. First, Benders reformulations seem to be intrinsically more efficient compared to branch-and-cut (GRB). Specifically, our best implementation (B-F), attains an average gap which is approximately 30 times lower, and consumes about 70% of GRB's CPU time (80% for the $\mathbb{R}$ instances). This performance difference has been observed for other problems as well, such as facility location (Fischetti et al., 2016b), suggesting that a modern implementation of Benders decomposition can be a superior alternative to an off-the-shelf, state-of-the-art solver. Second, with respect to the basic Benders implementation, we note that although it achieves a far better gap than branch-and-cut, it falls behind the more sophisticated implementations by a large margin. Third, when assessing the impact of PO cuts, we observe a non-trivial gap improvement between the basic Benders B-R and the basic PO Benders B-MW1 implementations, implying that the impact of adding PO cuts significantly enhances performance. However, these improvements come at a high CPU time cost, because two LPs are solved to generate every optimality cut, resulting in B-MW1 having the worst time performance across all methods. Fourth, implementation B-MW2, which generates PO cuts using a single subproblem per iteration dominates B-MW1 in terms of both gap and CPU time performance. Finally, B-FSZ is the best-performing implementation, having clearly the lowest gap and CPU time. In particular, it was able to solve optimally 99 (54 for R) instances that GRB failed to solve to optimality, and 58 (39 for R) instances that no other algorithm solved optimally. The ability of this model to incorporate the current flow cost in the cut-generating subproblem and to generate cuts that unify feasibility and optimality leads to important improvements over the Magnanti-Wong implementations. Finally, the breakdown per original network shows that the USC30 instances can all be solved to near-optimality rather easily, while JBH50 and USC53 are more challenging. In addition, problems with more periods are generally more challenging to solve, but interestingly B-FSZ seems to attain a better gap performance for longer horizons, using about the same amount of CPU time.

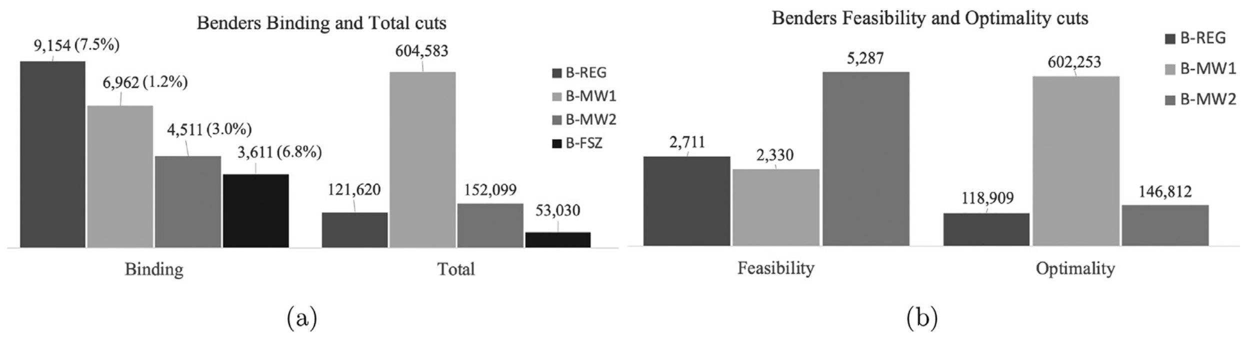

Fig. 1. Cuts generated by each implementation. Optimality and feasibility refer to the totals generated during branch-and-cut. Binding are the cuts binding at an optimal solution. Numbers in parentheses are proportions of total generated cuts.

In an attempt to further improve the performance of our algorithms for hard instances, we have experimented with adding cuts at the master problem root node prior to branching by using an in-out scheme as described in Fischetti et al. (2016a). To demonstrate the potential of this scheme, we considered the ten hardest instances defined as those with the largest integrality gap when solved by the B-FSZ method, and benchmarked B-FSZ with and without the in-out cuts. For these instances, the average root node lower bound improvement is 9.13%, while the overall average gap improves from 9.65% to 4.42%. Detailed results are reported in Appendix D.

Generated Benders cuts. On further analysis, it is interesting to investigate the number and type of cuts each Benders implementation generates. To this end, Fig. 1 shows the average number of binding and total cuts (Panel (a)) and feasibility and optimality cuts (Panel (b)) each algorithm generated in a subset of 306 Pazour instances for which all algorithms could find an optimal solution. Note that we do not report a breakdown into feasibility and optimality cuts for the B-FSZ method, because its implementation does not use the feasibility detection subroutine which the other methods use, and therefore the numbers of feasibility and optimality cuts it generates are not directly comparable.

The average number of Benders cuts generated over a large number of instances may lead to some notable insights. First, in terms of cuts binding at an optimal solution, regular Benders is outperformed by all other algorithms, followed by B-MW1, B-MW2 and B-F. The regular implementation MW1, which solves two LPs to generate a single PO cut, generates a very large number of optimality cuts, and the smallest number of feasibility cuts. Interestingly, B-FSZ generates the smallest number of binding and total optimality cuts and the second largest proportion of binding cuts, suggesting that the generated cuts are more likely to be active at an optimal solution. This experiment underlines that there may be a considerable spectrum when it comes to the performance variability of PO cuts. In particular, generating the cuts by solving LPs could result in strong cuts, since there exists some variability embedded in the pivoting rules that return one among multiple optimal solutions. An interesting direction for future research is to devise a mechanism that exploits this variability in a systematic way.

6.3. Solution analysis

A focal question pertinent to multi-period settings is how the inclusion of the multi-period structure influences the characteristics of the resulting solutions. Specifically, we investigate the timing of the arc opening decisions, the variable versus fixed costs composition and commodity flow changes over the horizon. To this end, we partition the problem horizon in early, middle and late periods, and report aggregate statistics for each segment. This partition is made so that each segment captures $|T|/3$ periods, with the middle segment capturing more periods when rounding is necessary. Since we did not observe any major differences in the behavior of capacitated or uncapacitated instances, we report our analysis for capacitated instances of the Pazour dataset, based on the solutions we obtained from our integrated Lagrange Relaxation algorithm. For brevity, we only summarize conclusions that (i) have an interesting interpretation and (ii) are persistent across capacitated and uncapacitated instances. In particular, we do not report results for different correlations and routing cost modes because the differences are small across the corresponding configurations. The complete results of our solution analysis study, including the uncapacitated instances and the R instances, are available from the authors upon request.

Arc opening timing. The first matter of interest is the timing of arc opening. Overall, 80% of the arcs used are opened during early, 7% during middle and 13% during late periods. Despite the lower demand in early periods, the fact that all commodities have to be routed from their origins to their destinations induces the majority of the arcs to be opened early on. Yet another factor that could influence early opening is that the original networks are very sparse and have a relatively large number of commodities, and therefore the number of feasible and cost-efficient paths per commodity may be limited. In addition, since commodity demands increase through the horizon, it becomes more important to contain the routing cost, resulting in relatively more arcs opening in late periods than in middle periods. When considering arc opening separately for different horizon lengths, however, Fig. 2 shows that for an horizon of five periods there are significantly fewer arcs opened early, and more arcs opened late. This suggests that models with too short horizons used as approximations to long horizon problems could underestimate the number of arcs to be opened early, leading to suboptimal solutions. This is not

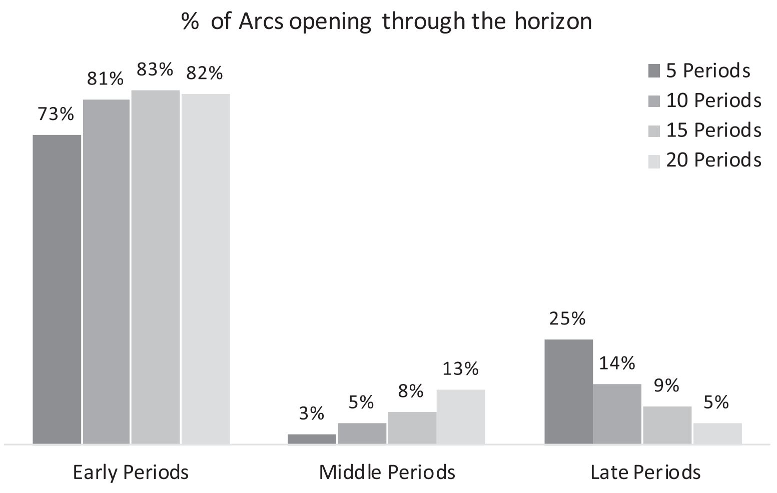

Fig. 2. Proportion of arcs opening over early, middle and late periods.

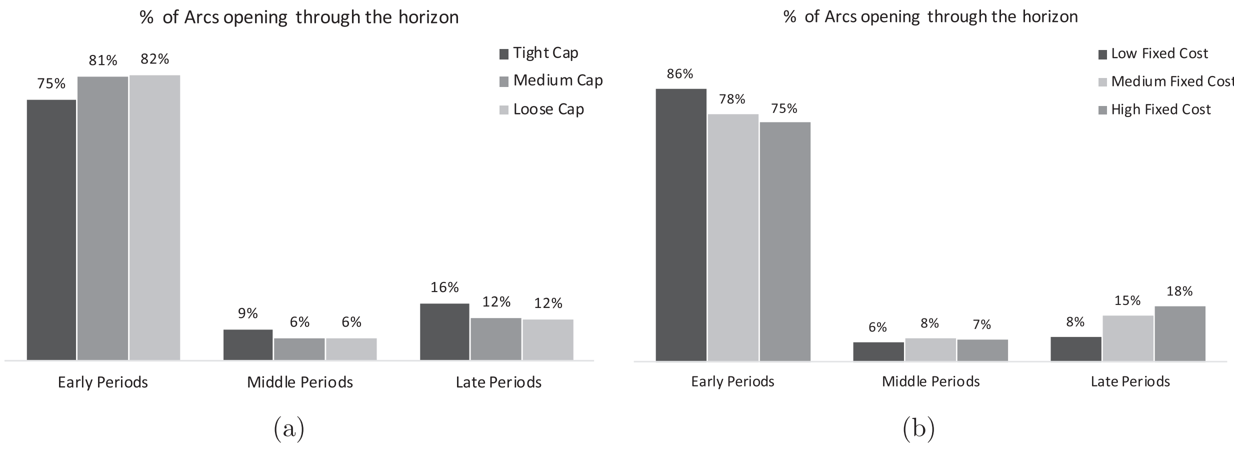

Fig. 3. Proportion of arcs opening per capacity (a) and fixed cost level (b).

the case for horizons with 10 or more periods, where the proportion of arcs opened early seems to stabilize. In addition, the more periods are considered, the more arcs are opened in middle periods, leading to a "smoother" opening plan, with a decreasing number of arcs opening as we approach the end of the horizon. This is also the case with the R instances, which extend to 80 periods.

It is also interesting to investigate the impact that capacity tightness and cost ratio have on arc opening. To this end, Fig. 3 shows the proportion of opened arcs for each capacity and fixed cost level. The impact of capacity tightness is more pronounced over middle and late periods, where more arcs need to be opened when capacity is tight, leaving fewer arcs to be opened early. In other words, the higher percentage of arcs that are opened late is a consequence of the overall demand increase, coupled with tight network capacities. When it comes to fixed costs, it is more beneficial to defer arc opening when the fixed cost is high relative to the variable cost. Interestingly, this does not make a significant difference in middle periods, where commodity demand is linear, but a significant difference in early and late periods.

Cost composition. Another matter of interest is how the arc opening decisions influence the proportion of the resulting fixed versus variable cost over time. To this end, recall that fixed costs decrease linearly as we approach the end of the horizon, while commodity routing costs remain constant and demand increases. The combined effect of those trends is that the proportion of fixed costs decreases from 54% in early to 5% and 3%, in middle and late periods, respectively. In other words, after the initial arc opening phase, the predominant cost results from routing the commodities through the network. However, this composition depends heavily on the magnitude of fixed costs, as Fig. 4 shows.

Reflecting on Fig. 4, we observe that high fixed costs account for 70% of the overall cost in early periods despite the observation that fewer arcs are opened in this case, as shown by Fig. 3 b. Taken together, when the opening cost is high, as is typically the case for constructing rail segments, it seems beneficial to selectively construct certain arcs early, even if they bear a high initial cost, since the increase in demand can be tackled in later periods, where fixed costs are less expensive.

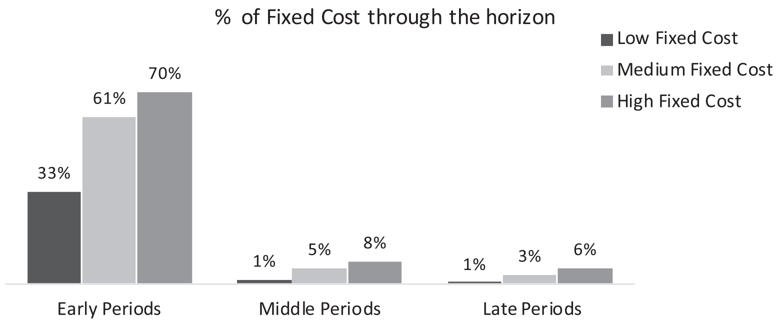

Fig. 4. Fixed cost composition per horizon segment.

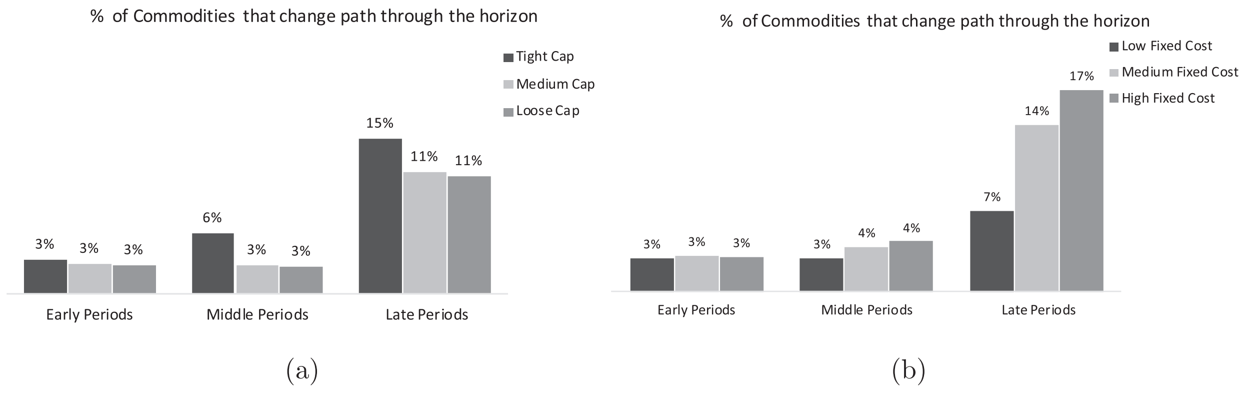

Fig. 5. Proportion of rerouted commodities per capacity (a) and fixed cost level (b).

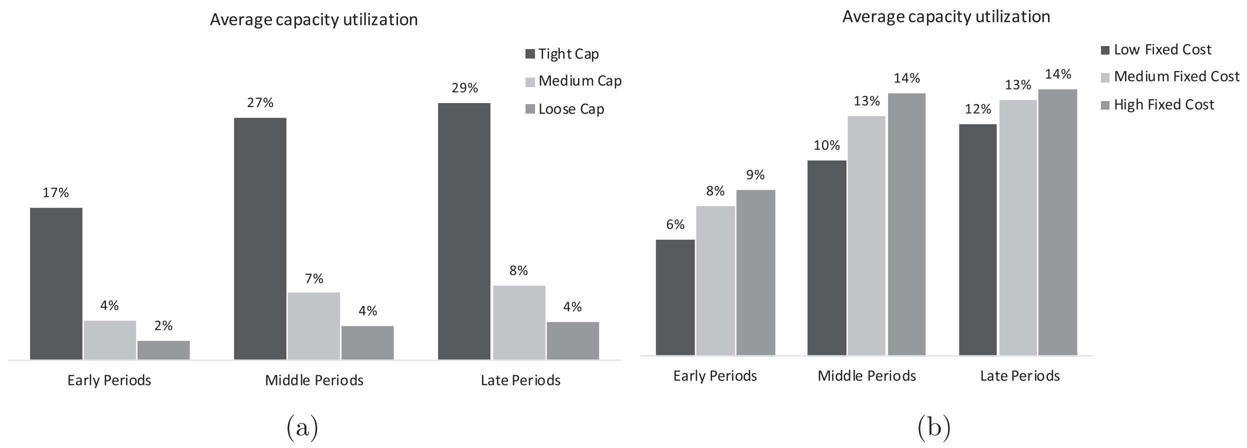

Fig. 6. Average capacity utilization per capacity (a) and fixed cost level (b).

Commodity Rerouting. Another consequence of increasing commodity demands is that more commodities change their routes in late periods, compared to their initial routes. Specifically, in early and middle periods, only 3% and 4% of commodities are rerouted, respectively, compared to 12% in late periods. Fig. 5 shows that this increases to 15% when capacities are tight and to 17% when fixed costs are high. This latter case of high fixed costs reflects a higher proportion of arcs opening late, as suggested by panel b of Fig. 2, which in turn allows more commodities to reroute their original paths.

Capacity Utilization. We conclude this part by commenting on the capacity utilization, defined as the average proportion of capacity that is being used by routing commodities across opened arcs and horizon segments. The average capacity utilization is 8%, 12% and 13% for early, middle and late periods, respectively. Such low utilization is likely to be related to the nature of the underlying networks, which are quite sparse and have a large number of commodities with diverse origins and destinations. This diversity implies that it may be necessary to open specific arcs only to route a small number of commodities, thereby reducing the average capacity utilization. In contrast, the R instances, which are randomly generated and significantly denser, reach utilization rates of 59%, 65% and 70% for early, middle and late periods, respectively. Fig. 6 shows how capacity utilization varies among instances with varying capacity and fixed cost levels.

The disproportionate impact of tight capacities is depicted in panel Fig. 6a, while panel Fig. 6b shows that the higher the fixed cost, the more beneficial it is to increase capacity utilization, suggesting that expensive arcs should be better utilized. However, note that the differences are small, and utilization practically stabilizes for middle and late periods. To interpret this finding, recall that in single-period problems the core trade-off is to balance (i) excessive fixed costs by opening a few key arcs through which many commodities can be routed with (ii) routing each commodity through its minimum cost paths. In a multi-period setting, however, fixed costs have a one-off impact, while routing costs are recurrent, and rise when demand is increasing. Therefore, for problems such as designing and expanding a freight rail network, the impact of arc construction decisions on the year-to-year operational costs can be detrimental, and a low capacity utilization should not be perceived as inefficient a priori. This is a key insight coming from the temporal interplay of opening and routing decisions, but research that utilizes actual cost data is required to validate this claim.

7. Extensions and future research

We introduce a multi-period extension of the multi-commodity network design problem that arises in an array of applications, and study its capacitated and uncapacitated variants. By taking advantage of problem structure, we develop heuristic and decomposition-based algorithms that exhibit better performance than a commercial, state-of-the-art branch-and-cut solver. For capacitated problems, Lagrange relaxation scales very well with problem size and delivers high quality lower and upper bounds across a large range of problem sizes. For uncapacitated problems, we employ a variety of Benders decomposition formulations and show that the best one solves very large instances to optimality. Our algorithms are tested on two sets of networks, for which we analyze the temporal characteristics of solutions.

A potential limitation of our study is that it assumes deterministic commodity demands. For freight rail applications, it is common to assume a target demand (Bärmann et al., 2017) which reflects anticipated growth and strategic targets. In other contexts, such as DHL's strategic service network design case (Cheung et al., 2001), deterministic network design may be combined with a simulation model to assess the network's operational characteristics and solution quality in a stochastic environment, although for such problems using a deterministic solution may lead to fewer consolidation opportunities (Lium et al., 2009). Further, deterministic solutions can be used to decide which arcs to open and when (Thapalia et al., 2012), and then solve the corresponding stochastic LPs. While conceptually simple, this heuristic may substantially improve the deterministic solution, by as much as 97% (Sun et al., 2017). For multi-period problems, where the routing costs are predominant over fixed costs, it is likely that such a method delivers good results.

There are several avenues for future research. For capacitated problems, volume-based branch-and-bound appears to be an approach worth exploring. Investigating the efficiency of well-established heuristics, such as large neighborhood search (Ropke and Pisinger, 2006), evolutionary algorithms (Paraskevopoulos et al., 2016), slope scaling (Crainic et al., 2004) or column and row generation (Katayama et al., 2009) can lead to further improved solutions. For uncapacitated problems, further improvements on the Benders cut generation mechanism can be investigated, particularly the incorporation of in-out schemes (Fischetti et al., 2016a) that have the potential to improve the root node lower bound. We hope that our study inspires future work on such problems.

Declaration of Competing Interest

No.

CRediT authorship contribution statement

Ioannis Fragkos: Conceptualization, Methodology, Software, Validation, Formal analysis, Data curation, Investigation, Writing - original draft. **Jean-François Cordeau:** Conceptualization, Methodology, Validation, Resources, Writing - original draft, Supervision, Funding acquisition. **Raf Jans:** Conceptualization, Methodology, Validation, Resources, Writing - original draft, Supervision, Funding acquisition.

Acknowledgment

This research was partly funded by the Fonds de recherche du Quèbec - Nature et technologies. This support is gratefully acknowledged. We also want to thank Dr. Jennifer Pazour, who provided us with the networks JBH50, USC30 and USC53.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.trb.2020.12.002.

References

Adulyasak, Y., Cordeau, J.-F., Jans, R., 2015. Benders decomposition for production routing under demand uncertainty. Oper. Res. 63 (4), 851–867.

Alvelos, F., de Carvalho, J., 2007. An extended model and a column generation algorithm for the planar multicommodity flow problem. Networks 50 (1), 3–16.

Atamtürk, A., 2001. Flow pack facets of the single node fixed-charge flow polytope. Oper. Res. Lett. 29 (3), 107–114.

Atamtürk, A., 2002. On capacitated network design cut–set polyhedra. Math. Program. 92 (3), 425–437.

Atamtürk, A., Rajan, D., 2002. On splittable and unsplittable flow capacitated network design arc–set polyhedra. Math. Program. 92 (2), 315–333.

Bai, L., Rubin, P., 2009. Combinatorial Benders cuts for the minimum tollbooth problem. Oper. Res. 57 (6), 1510–1522.

Balakrishnan, A., Magnanti, T., Wong, R., 1989. A dual-ascent procedure for large-scale uncapacitated network design. Oper. Res. 37 (5), 716–740.

Barahona, F., Anbil, R., 2000. The volume algorithm: producing primal solutions with a subgradient method. Math. Program. 87 (3), 385–399.

Bärmann, A., Martin, A., Schülldorf, H., 2017. A decomposition method for multiperiod railway network expansion–with a case study for Germany. Transp. Sci. 51 (4), 1102–1121.

Bienstock, D., Günlük, O., 1996. Capacitated network design-polyhedral structure and computation. INFORMS J. Comput. 8 (3), 243–259.

Blanco, V., Puerto, J., Ramos, A., 2011. Expanding the Spanish high-speed railway network. Omega 39 (2), 138–150.

Cheung, W., Leung, L., Wong, Y., 2001. Strategic service network design for DHL Hong Kong. Interfaces 31 (4), 1–14.

Chouman, M., Crainic, T., Gendron, B., 2009. A cutting-plane algorithm for multicommodity capacitated fixed-charge network design. CIRRELT Montreal.

Chouman, M., Crainic, T., Gendron, B., 2017. Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. Transp. Sci. 51 (2), 650–667.

Cordeau, J.-F., Soumis, F., Desrosiers, J., 2001. Simultaneous assignment of locomotives and cars to passenger trains. Oper. Res. 49 (4), 531–548.

Costa, A., Cordeau, J.-F., Gendron, B., 2009. Benders, metric and cutset inequalities for multicommodity capacitated network design. Comput Optim Appl 42 (3), 371–392.

Crainic, T., Frangioni, A., Gendron, B., 2001. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. Discrete Appl. Math. 112 (1), 73–99.

Crainic, T., Gendron, B., Hernu, G., 2004. A slope scaling/Lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. J. Heuristics 10 (5), 525–545.

Cruz, F., Smith, J., Mateus, G., 1998. Solving to optimality the uncapacitated fixed-charge network flow problem. Comput. Oper. Res. 25 (1), 67–81.

De Araujo, S.A., De Reyck, B., Degraeve, Z., Fragkos, I., Jans, R., 2015. Period decompositions for the capacitated lot sizing problem with setup times. INFORMS J. Comput. 27 (3), 431–448.

Fischetti, M., Ljubić, I., Sinnl, M., 2016. Benders decomposition without separability: acomputational study for capacitated facility location problems. Eur. J. Oper. Res. 253 (3), 557–569.

Fischetti, M., Ljubić, I., Sinnl, M., 2016. Redesigning Benders decomposition for large-scale facility location. Manage. Sci. 63 (7), 2146–2162.

Fischetti, M., Lodi, A., 2003. Local branching. Math. Program. 98 (1–3), 23–47.

Fischetti, M., Salvagnin, D., Zanette, A., 2010. A note on the selection of Benders' cuts. Math. Program. 124 (1–2), 175–182.

Fisher, M., 1981. The Lagrangian relaxation method for solving integer programming problems. Manage. Sci. 27 (1), 1–18.

Fontaine, P., Minner, S., 2014. Benders decomposition for discrete–continuous linear bilevel problems with application to traffic network design. Transp. Res. Part B 70, 163–172.

Frangioni, A., Gendron, B., 2009. 0–1 Reformulations of the multicommodity capacitated network design problem. Discrete Appl. Math. 157 (6), 1229–1241.

Frangioni, A., Gendron, B., 2013. A stabilized structured Dantzig–Wolfe decomposition method. Math. Program. 140 (1), 45–76.

Gao, Z., Wu, J., Sun, H., 2005. Solution algorithm for the bi-level discrete network design problem. Transp. Res. Part B 39 (6), 479–495.

Gendron, B., Crainic, T., 1994. Relaxations for Multicommodity Capacitated Network Design Problems. CRT Technical Report 965. Université de Montréal, Canada.

Ghamlouche, I., Crainic, T., Gendreau, M., 2003. Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. Oper. Res. 51 (4), 655–667.

Günlük, O., 1999. A branch-and-cut algorithm for capacitated network design problems. Math. Program. 86 (1), 17–39.

Holmberg, K., Yuan, D., 1998. A Lagrangean approach to network design problems. Int. Trans. Oper. Res. 5 (6), 529–539.

Holmberg, K., Yuan, D., 2000. A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. Oper. Res. 48 (3), 461–481.

Hooghiemstra, J., Kroon, L., Odijk, M., Salomon, M., Zwaneveld, P., 1999. Decision support systems support the search for win-win solutions in railway network design. Interfaces 29 (2), 15–32.

Idzikowski, F., Orlowski, S., Raack, C., Woesner, H., Wolisz, A., 2011. Dynamic routing at different layers in IP-over-WDM networks–maximizing energy savings. Opt. Switching Netw. 8 (3), 181–200.

Jena, S., Cordeau, J.-F., Gendron, B., 2015. Dynamic facility location with generalized modular capacities. Transp. Sci. 49 (3), 484–499.

Kalinowski, T., Matsypura, D., Savelsbergh, M., 2015. Incremental network design with maximum flows. Eur. J. Oper. Res. 242 (1), 51–62.

Katayama, N., Chen, M., Kubo, M., 2009. A capacity scaling heuristic for the multicommodity capacitated network design problem. J. Comput. Appl. Math. 232 (1), 90–101.

Kim, D., Pardalos, P., 1999. A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. Oper. Res. Lett. 24 (4), 195–203.

Lai, Y., Shih, M., 2013. A stochastic multi-period investment selection model to optimize strategic railway capacity planning. J. Adv. Transp. 47 (3), 281–296.

Lee, D., Dong, M., 2008. A heuristic approach to logistics network design for end-of-lease computer products recovery. Transp. Res. Part E 44 (3), 455–474.

Liu, H., Wang, D., 2015. Global optimization method for network design problem with stochastic user equilibrium. Transportation Research Part B: Methodological 72, 20–39.

Lium, A., Crainic, T., Wallace, S., 2009. A study of demand stochasticity in service network design. Transp. Sci. 43 (2), 144–157.

Luathep, P., Sumalee, A., Lam, W., Li, Z., Lo, H., 2011. Global optimization method for mixed transportation network design problem: a mixed-integer linear programming approach. Transp. Res. Part B 45 (5), 808–827.

Magnanti, T., Wong, R., 1981. Accelerating Benders decomposition: algorithmic enhancement and model selection criteria. Oper. Res. 29 (3), 464–484.

Magnanti, T., Wong, R., 1984. Network design and transportation planning: models and algorithms. Transp. Sci. 18 (1), 1–55.

Magnanti, T.L., Mireault, P., R. T. Wong, R.T., 1986. Tailoring Benders decomposition for uncapacitated network design. In: Netflow at Pisa. Springer, pp. 112–154.

Marin, A., Jaramillo, P., 2008. Urban rapid transit network capacity expansion. Eur. J. Oper. Res. 191 (1), 45–60.

Minoux, M., 1989. Networks synthesis and optimum network design problems: models, solution methods and applications. Networks 19 (3), 313–360.

Papadakos, N., 2008. Practical enhancements to the magnanti–wong method. Oper. Res. Lett. 36 (4), 444–449.

Papadimitriou, D., Fortz, B., 2015. A rolling horizon heuristic for the multiperiod network design and routing problem. Networks 66 (4), 364–379.

Paraskevopoulos, D., Bektaş, T., Crainic, T., Potts, C., 2016. A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. Eur. J. Oper. Res. 253 (2), 265–279.

Pazour, J., Meller, R., Pohl, L., 2010. A model to design a national high-speed rail network for freight distribution. Transp. Res. Part A 44 (3), 119–135.

Petersen, E., Taylor, A., 2001. An investment planning model for a new north-central railway in Brazil. Transp. Rese. Part A 35 (9), 847–862.

Raack, C., Koster, A., Orlowski, S., Wessäly, R., 2011. On cut-based inequalities for capacitated network design polyhedra. Networks 57 (2), 141–156.

Rahmaniani, R., Crainic, T., Gendreau, M., Rei, W., 2016. The Benders decomposition algorithm: a literature review. Eur. J. Oper. Res. 259 (3), 801–817.

Randazzo, C., Luna, H., 2001. A comparison of optimal methods for local access uncapacitated network design. Ann. Oper. Res. 106 (1–4), 263–286.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp. Sci. 40 (4), 455–472.
Schrijver, A., 1998. Theory of Linear and Integer Programming. John Wiley & Sons, New York, NY.
Şeref, O., Ahuja, R., Orlin, J., 2009. Incremental network optimization: theory and algorithms. Oper. Res. 57 (3), 586–594.
Sun, C., Wallace, S., Luo, L., 2017. Stochastic multi-commodity network design: the quality of deterministic solutions. Oper. Res. Lett. 45 (3), 266–268.
Thapalia, B., Wallace, S., Kaut, M., Crainic, T., 2012. Single source single-commodity stochastic network design. Comput. Manage. Sci. 9 (1), 139–160.
Tong, L., Zhou, X., Miller, H., 2015. Transportation network design for maximizing space–time accessibility. Transp. Res. Part B 81, 555–576.
Ukkusuri, S., Patil, G., 2009. Multi-period transportation network design under demand uncertainty. Transp. Res. Part B 43 (6), 625–642.
Wang, S., Meng, Q., Yang, H., 2013. Global optimization methods for the discrete network design problem. Transp. Res. Part B 50, 42–60.
Yaghini, M., Karimi, M., Rahbar, M., Sharifitabar, M., 2014. A cutting-plane neighborhood structure for fixed-charge capacitated multicommodity network design problem. INFORMS J. Comput. 27 (1), 48–58.
Yang, H., Bell, H., Michael, G., 1998. Models and algorithms for road network design: a review and some new developments. Transp. Rev. 18 (3), 257–278.
Zetina, C., Contreras, I., Cordeau, J., 2019. Profit-oriented fixed-charge network design with elastic demand. Transp. Res. Part B 127, 1–19.
Zhu, E., Crainic, T., Gendreau, M., 2014. Scheduled service network design for freight rail transportation. Oper. Res. 62 (2), 383–400.