

Solving Unsplittable Network Flow Problems with Decision Diagrams

Hosseinali Salemi, Danial Davarnia

Department of Industrial and Manufacturing Systems Engineering, Iowa State University, Ames, IA 50011,
hsalemi@iastate.edu, davarnia@iastate.edu

In unsplittable network flow problems, certain nodes must satisfy a combinatorial requirement that the incoming arc flows cannot be split or merged when routed through outgoing arcs. This so-called *no-split no-merge* requirement arises in unit train scheduling where train consists should remain intact at stations that lack necessary equipment and manpower to attach/detach them. Solving the unsplittable network flow problems with standard mixed-integer programming formulations is computationally difficult due to the large number of binary variables needed to determine matching pairs between incoming and outgoing arcs of nodes with no-split no-merge constraint. In this paper, we study a stochastic variant of the unit train scheduling problem where the demand is uncertain. We develop a novel decision diagram (DD)-based framework that decomposes the underlying two-stage formulation into a master problem that contains the combinatorial requirements, and a subproblem that models a continuous network flow problem. The master problem is modeled by a DD in a transformed space of variables with a smaller dimension, leading to a substantial improvement in solution time. Similarly to the Benders decomposition technique, the subproblems output cutting planes that are used to refine the master DD. Computational experiments show a significant improvement in solution time of the DD framework compared with that of standard methods.

Key words: Decision Diagrams; Network Optimization; Mixed Integer Programs; Unit Trains; Transportation

History:

1. Introduction

Over the past several decades, rail freight transportation has continued to grow as the prime means of transportation for high-volume commodities. Advantages of rail transportation include reliability, safety, cost-efficiency and environmental-sustainability as compared with alternative methods of transportation. In terms of scale, the rail network accounted for 27.2 percent of U.S. freight shipment by ton-miles in 2018 (Furchtgott-Roth et al. 2021); see Figure 1. The Federal Highway Administration estimates that the total U.S. freight shipments will be 24.1 billion tons in 2040, a 30 percent increase from the 2018 total transportation of 18.6 billion tons. With the purpose of meeting such market growth, America’s freight railway companies have invested nearly

\$740 billion on capital expenditures and maintenance from 1980 to 2020 (Association of American Railroads 2021).

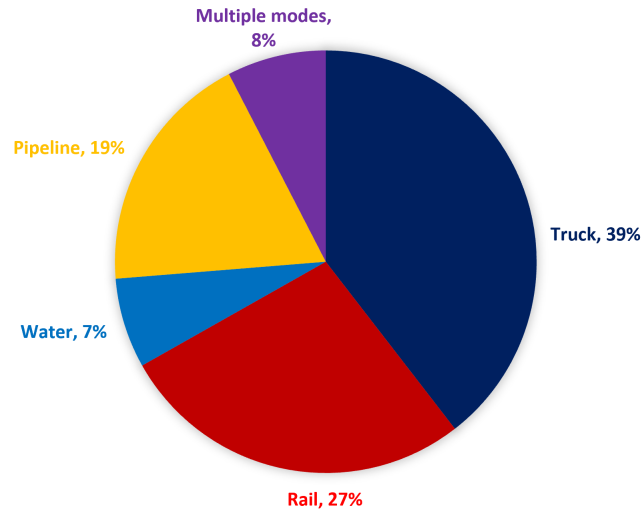


Figure 1 Pie chart for ton-miles of freight shipments by mode within the U.S. in 2018. Multiple modes includes mail. Air and truck-air with the share of 0.1% are omitted.

To reduce rail freight transportation costs and shipment delays, railroad companies offer *unit train* services for carrying high-volume products. Unit trains haul a single type freight in a way that no car is attached or detached while the cargo train is on its way from an origin to a destination, except in specific locations that are equipped with required manpower and machinery. These trains usually operate all day, use dedicated equipment, and can be loaded/unloaded in 24 hours. They are known to be one of the fastest and most efficient means of railroad transportation. (Association of American Railroads 2021). Traditionally, unit trains are used to carry bulk cargo such as coal, grain, cement, and rock. Bulk liquids like crude oil and food such as wheat and corn are also shipped by unit trains. According to the Federal Railroad Administration data, bulk commodities account for 91 percent of the U.S. railroad freights. Approximately all coal shipped through railways in the U.S. are transported by unit trains. Moreover, these trains contribute significantly to the shipping process of crude oil as each unit train is capable of carrying 85,000 barrels (Association of American Railroads 2021). In an operational level, the core unit train model can be described as follows. Given a set of supply, intermediate, and demand locations in a railroad network, the unit train scheduling problem seeks to find optimal routes for unit trains to send flows from supply to demand points with the objective of minimizing the total transportation cost while meeting demand of customers, respecting capacities of tracks, and satisfying no-car attaching/detaching requirements in specific locations. As a result, designing *blocking plans* to determine locations

where cars need to be switched between trains is irrelevant in this problem, unlike scheduling other types of trains (Davarnia et al. 2019).

Despite the significance of unit train scheduling, exact optimization approaches to solve associated problems are scarce, partially due to their structural complexities. One of the main challenges in modeling unit trains is the requirement that the train consists must remain intact when passing through stations that lack necessary busting/formation equipment. In optimization, this requirement is referred to as *no-split no-merge* (NSNM), which guarantees that the flows entering to or exiting from certain nodes of the unit train network cannot be split or merged. Incorporating this requirement into typical transportation network models yields the so-called *generalized unsplittable flow problem* (GUFP), where the objective is to determine the minimum-cost unit train schedules that satisfy the given demand. Numerous studies have shown that considering deterministic demands might result in the complete failure of the transportation scheduling (Demir et al. 2016, Layeb et al. 2018), motivating the study of stochastic variants of the unit train scheduling problems where the demand is uncertain. As a result, in this paper, we consider a stochastic variant of the GUFP, referred to SGUFP, that is modeled as a two-stage optimization problem. The first stage decides a matching between the incoming and outgoing arcs of the nodes of the railroad network, and the second stage determines the amount of flow that should be sent through the matching arcs of the network to satisfy the uncertain demand represented by a number of demand scenarios. We propose a novel exact solution framework to solve this problem in the operational level.

Our proposed methodology is based on *decision diagrams* (DDs), which are compact graphical data structures. DDs were initially introduced to represent boolean functions with applications in circuit design. Over the past decade, researchers have successfully extended DDs domain by developing DD-based algorithms to solve discrete optimization problems in different areas of application. Because of its structural limitation to model integer programs only, DDs have never been used to solve transportation problems that inherently include continuous variables. In this paper, we extend the application scope of DDs by introducing a novel framework that is capable of modeling network problems with both integer and continuous components as in the SGUFP.

1.1. Literature Review on Train Scheduling

Many variants of train routing and scheduling problems with different objective functions and set of constraints under deterministic and stochastic conditions have been introduced and vastly studied in the literature; see surveys by Cordeau, Toth, and Vigo (1998), Harrod and Gorman (2010), Lusby et al. (2011), Cacchiani and Toth (2012), and Turner et al. (2016) for different problems classifications and structures. Mixed integer linear and nonlinear programming formulations are among the most frequent exact approaches to model different classes of these problems (Jovanović and Harker 1991, Huntley et al. 1995, Sherali and Suharko 1998, Lawley et al.

2008, Haahr and Lusby 2017, Davarnia et al. 2019). Proposed solution techniques include but are not limited to branch-and-bound methods (Jovanović and Harker 1991, Fuchsberger and Lüthi 2007), branch-and-cut frameworks (Zwaneveld, Kroon, and Van Hoesel 2001, Ceselli et al. 2008), branch-and-price approaches (Lusby 2008, Lin and Kwan 2016), graph coloring algorithms (Cornelsen and Di Stefano 2007), and heuristics (Carey and Crawford 2007, Liu and Kozan 2011, İċyüz et al. 2016). Rolling stock scheduling (Abbink et al. 2004, Alfieri et al. 2006, Haahr et al. 2016, Borndörfer et al. 2016) that assigns rolling stocks to a given timetable, and crew scheduling (Kwan 2011, Shen et al. 2013, Heil, Hoffmann, and Buscher 2020) that covers train activities by assigning crews to the associated operations are other major problems arising in the area of railroad planning.

Due to the inherent uncertainty in different types of train scheduling and routing problems, many researchers have studied stochastic variants of the problems where the supply/demand is considered to be uncertain. Jordan and Turnquist (1983) propose a model for railroad car distribution where supply and demand of cars are uncertain. Jin et al. (2019) study a chance-constrained programming model for the train stop planning problem under stochastic demand. Ying, Chow, and Chin (2020) propose a deep reinforcement learning approach for train scheduling where the passenger demand is uncertain. Recently, Gong et al. (2021) propose a stochastic optimization method to solve a train timetabling problem with uncertain passenger demand. Also see works by Meng and Zhou (2011), Quaglietta, Corman, and Goverde (2013), Larsen et al. (2014) that consider train dispatching problems under stochastic environments.

In the context of unit train scheduling, Lawley et al. (2008) study a time-space network flow model to schedule bulk railroad deliveries for unit trains. In their model, the authors consider characteristics of underlying rail network, demands of customers, and capacities of tracks, stations, and loading/unloading requirements. They propose a mixed integer programming (MIP) formulation that maximizes the demand satisfaction while minimizing the waiting time at stations. Lin and Kwan (2014) (cf. Lin and Kwan (2016)) propose a model for a train scheduling problem that is capable to capture locations where coupling/decoupling is forbidden. They develop a branch-and-price algorithm inspired by column generation to solve the associated problem. Lin and Kwan (2018) also propose a heuristic branch-and-bound approach to decrease coupling/decoupling redundancy. İċyüz et al. (2016) study the problem of planning coal unit trains that includes train formation, routing, and scheduling. As noted by the authors, their proposed MIP formulation fails to solve the problem directly due to its large size. As a remedy, they develop a time-efficient heuristic that produces good quality solutions. More recently, Davarnia et al. (2019) introduce and study the GUFP with application to unit train scheduling. In particular, the authors show how to impose NSNM restrictions in network optimization problems. They present a polyhedral study and propose a MIP formulation to model a stylized variant of the unit train scheduling problem. In the present paper, we use their formulation (see section 3.1) as a basis for our solution framework.

The unsplittable flow problem (UFP) was first introduced by Kleinberg (1996) as a generalization of the disjoint path problem. Given a network with capacities for arcs and a set of source-terminal vertex pairs with associated demands and rewards, the objective in the UFP is to maximize the total revenue by selecting a subset of source-terminal pairs and routing flows through a *single* path for each of them to satisfy the associated demand. In the GUFP, however, there can exist nodes that do not need to respect the NSNM requirement, and demands can be satisfied by passing flows through multiple paths. It is well-known that different variants of UFP are NP-hard (Baier, Köhler, and Skutella 2005, Kolman and Scheideler 2006, Chakrabarti et al. 2007). Since its introduction, the UFP structure has been used in different areas of application, from bandwidth allocation in heterogeneous networks (Kolman and Scheideler 2006), to survivable connection-oriented networks (Walkowiak 2006), and virtual circuit routing problems (Hu, Lan, and Wan 2009). Considering the hardness of the problem, approximation algorithms have been a common technique to tackle different variants of the UFP in the literature (Baier, Köhler, and Skutella 2005, Chakrabarti et al. 2007).

1.2. Literature Review on Decision Diagrams

DDs are directed acyclic graphs with a source and a terminal node where each source-terminal path encodes a feasible solution to an optimization problem. In DDs, each layer from the source to the terminal represents a decision variable where labels of arcs show their values. Hadžić and Hooker (2006) proposed to use DDs to model the feasible region of a discrete optimization problem and used it for postoptimality analysis. Later, Andersen et al. (2007) presented relaxed DDs to circumvent the exponential growth rate in the DD size when modeling large discrete optimization problems. Bergman et al. (2016b) introduced a branch-and-bound algorithm that iteratively uses relaxed and restricted DDs to find optimal solution. The literature contains many successful utilization of DDs in different domains; see works by Bergman and Cire (2018), Serra and Hooker (2019), Davarnia and Van Hoesve (2020), Gonzalez et al. (2020), and Hosseininasab and Van Hoesve (2021) for some examples.

Until recently, applications of DDs were limited to discrete problems, and the question on how to use DDs in solving optimization problems with continuous variables was unanswered. To address this limitation, Davarnia (2021) proposed a technique called arc-reduction that generates a DD that represents a relaxation of the underlying continuous problem. In a follow-up work, Salemi and Davarnia (2022a) established necessary and sufficient conditions for a general MIP to be representable by DDs. They showed that a bounded MIP can be remodeled and solved with DDs through employing a specialized Benders decomposition technique. In this paper, we build on this framework to design a novel DD-based methodology to solve the SGUFP.

1.3. Contributions

While there are several studies in the literature dedicated to the unit train problem, exact methodologies that provide a rigorous treatment of the NSNM requirement at the heart of unit train models are scarce. In this paper, we design a novel exact DD-based framework to solve the SGUFP, as a more realistic and more challenging variant of this problem class. To our knowledge, this is the first work that studies SGUFP from an exact perspective, and the first application of DDs to a transportation problem. Our proposed framework formulates the problem in a transformed space of variables, which has a smaller dimension compared to the standard MIP formulations of the SGUFP. This presentation mitigates the computational difficulties stemmed from the MIP formulation size, providing a viable solution approach for large-scale network problems. The core principles of our DD framework can also be used to model other transportation problems with similar structure, as an alternative to traditional network optimization techniques.

The remainder of this paper is organized as follows. In Section 2 we provide basic definitions and a brief overview on discrete and continuous DD models, including the DD-BD method to solve bounded MIPs. In Section 3, we adapt the DD-BD method to solve the SGUFP. We propose algorithms to construct exact and relaxed DDs to solve the problem in a transformed space. Section 4 presents computational experiments to evaluate the performance of the DD-BD method for the SGUFP. We give concluding remarks in Section 5.

2. Background on DDs

In this section, we present basic definitions and results relevant to our DD analysis.

2.1. Overview

A DD $\mathcal{D} = (\mathcal{U}, \mathcal{A}, l)$ with node set \mathcal{U} , arc set \mathcal{A} , and arc label mapping $l : \mathcal{A} \rightarrow \mathbb{R}$ is a directed acyclic graph with $n \in \mathbb{N}$ arc layers $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_n$, and $n + 1$ node layers $\mathcal{U}_1, \mathcal{U}_2, \dots, \mathcal{U}_{n+1}$. The node layers \mathcal{U}_1 and \mathcal{U}_{n+1} , with $|\mathcal{U}_1| = |\mathcal{U}_{n+1}| = 1$, contain the root r and the terminal t , respectively. In any arc layer $j \in [n] := \{1, 2, \dots, n\}$, an arc $(u, v) \in \mathcal{A}_j$ is directed from the tail node $u \in \mathcal{U}_j$ to the head node $v \in \mathcal{U}_{j+1}$. The *width* of \mathcal{D} is defined as the size of its largest \mathcal{U}_j . DDs can model a bounded integer set $\mathcal{P} \subseteq \mathbb{Z}^n$ in such a way that each r - t arc-sequence (path) of the form $(a_1, \dots, a_n) \in \mathcal{A}_1 \times \dots \times \mathcal{A}_n$ encodes a point $\mathbf{y} \in \mathcal{P}$ where $l(a_j) = y_j$ for $j \in [n]$, that is \mathbf{y} is an n -dimensional point in \mathcal{P} whose j -th coordinate is equal to the label value $l(a_j)$ of arc the a_j . For such a DD, we have $\mathcal{P} = \text{Sol}(\mathcal{D})$, where $\text{Sol}(\mathcal{D})$ denotes the finite collection of all r - t paths.

The graphical property of DDs can be exploited to optimize an objective function over a discrete set \mathcal{P} . To this end, DD arcs are weighted in such a way that the cumulative weight of an r - t path that encodes a solution $\mathbf{y} \in \mathcal{P}$ equals to the objective function value evaluated at \mathbf{y} . Then, a

shortest (resp. longest) r - t path for the underlying minimization (resp. maximization) problem is found, an operation that can be performed in polynomial time.

The construction of an *exact* DD as described above is computationally prohibitive due to the exponential growth rate of its size. To alleviate this difficulty, *relaxed* and *restricted* DDs are proposed to keep the size of DDs under control. In a relaxed DD, nodes are merged in such a way that the width of the resulting diagram is bounded by a predetermined width limit. This node-merging process ensures that all feasible solutions of the original set are encoded by a subset of all r - t paths in the resulting DD. Optimization over this relaxed DD provides a dual bound to the optimal solution of the original problems. In a restricted DD, the collection of all r - t paths of the DD encode a subset of the feasible solutions of the original set. Optimization over this restricted DD provides a primal bound to the optimal solution of the original problems. The restricted and relaxed DDs can be iteratively refined in a branch-and-bound scheme to find the optimal value of a problem through convergence of their primal and dual bounds. The following example illustrates an exact, relaxed and restricted DD for a discrete optimization problem.

EXAMPLE 1. Consider the discrete optimization problem $\max\{5y_1 + 10y_2 + 4y_3 \mid \mathbf{y} \in \mathcal{P}\}$ where $\mathcal{P} = \{(1, 0, 0), (1, 0, 1), (0, 1, 0), (0, 0, 1), (0, 0, 0)\}$. The exact DD \mathcal{D} with width 3 in Figure 2(a) models the feasible region \mathcal{P} . The weight of each arc $a \in \mathcal{A}_j$, for $j \in \{1, 2, 3\}$, shows the contribution of variable y_j 's value assignment to the objective function. The longest r - t path that encodes the optimal solution $(y_1^*, y_2^*, y_3^*) = (0, 1, 0)$ has length 10, which is the optimal value to the problem. By reducing the width limit to 2, we can build relaxed and restricted DDs for \mathcal{P} as follows. The relaxed DD $\overline{\mathcal{D}}$ in Figure 2(b) provides an upper bound to the optimal solution, where the longest path with length 14 is obtained by an infeasible point $(\overline{y}_1, \overline{y}_2, \overline{y}_3) = (0, 1, 1)$. Finally, the restricted DD $\underline{\mathcal{D}}$ in Figure 2(c) gives a lower bound to the optimal solution, where the longest path with length 9 encodes a feasible solution $(\underline{y}_1, \underline{y}_2, \underline{y}_3) = (1, 0, 1)$.

2.2. Continuous DD Models

While the framework described in the previous section can be applied to solve different classes of discrete optimization problems, its extension to model sets with continuous variables requires a fundamentally different approach. The reason that the traditional DD structure is not viable for continuous sets is that representing the domain of a continuous variable through arcs requires an infinite number of them, spanning all values within a continuous interval, which is structurally prohibitive in DD graphs. Fortunately, there is a way to overcome this obstacle by decomposing the underlying set into certain rectangular formations, which can in turn be represented through node-sequences in DDs. In what follows, we give an overview of these results as relevant to our analysis.

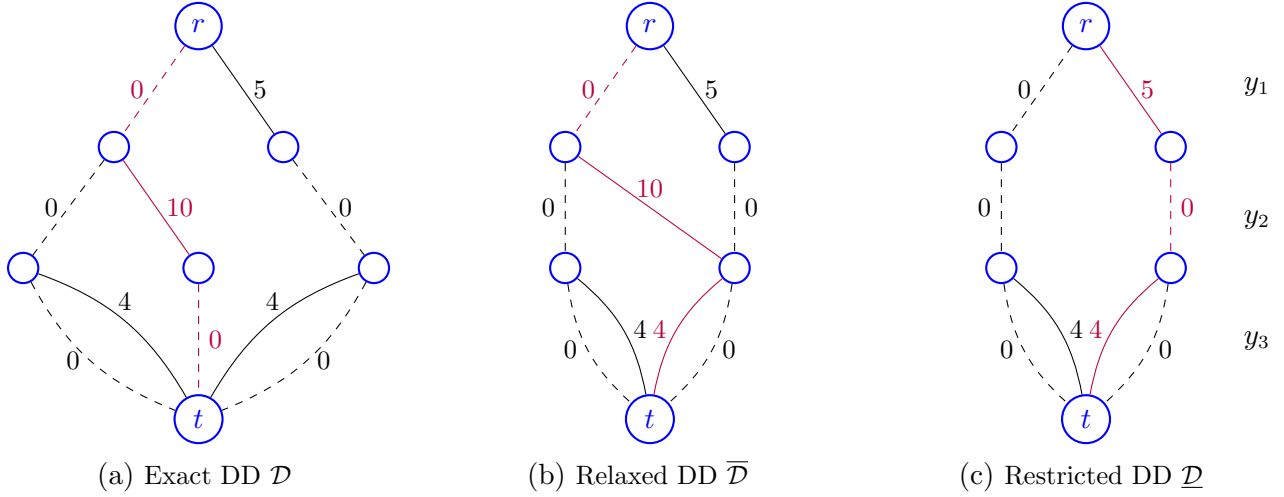


Figure 2 The exact, relaxed, and restricted DDs representing \mathcal{P} in Example 1. Solid and dotted arcs indicate one and zero arc labels, respectively. Numbers next to arcs represent weights.

Consider a bounded set $\mathcal{P} \subseteq \mathbb{R}^n$. Salemi and Davarnia (2022a) give necessary and sufficient conditions for \mathcal{P} to admit the desired rectangular decomposition. Such a set is said to be *DD-representable* w.r.t. a fixed index set $I \subseteq [n]$, as there exists a DD \mathcal{D} such that $\max\{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{P}\} = \max\{f(\mathbf{x}) \mid \mathbf{x} \in \text{Sol}(\mathcal{D})\}$ for every function $f(\mathbf{x})$ that is convex in the space of variables \mathbf{x}_I . A special case of DD-representable sets is given next.

PROPOSITION 1. *Any bounded mixed integer set of the form $\mathcal{P} \subseteq \mathbb{Z}^n \times \mathbb{R}$ is DD-representable w.r.t. $I = \{n+1\}$. \square*

This result gives rise to a novel DD-based framework to solve general bounded MIPs as outlined below. Consider a bounded MIP $\mathcal{H} := \max\{\mathbf{c}\mathbf{y} + \mathbf{d}\mathbf{x} \mid \mathbf{A}\mathbf{y} + \mathbf{G}\mathbf{x} \leq \mathbf{b}, \mathbf{y} \in \mathbb{Z}^n\}$. Using Benders decomposition (BD), formulation \mathcal{H} is equivalent to $\max_{\mathbf{y} \in \mathbb{Z}^n} \{\mathbf{c}\mathbf{y} + \max_{\mathbf{x}} \{\mathbf{d}\mathbf{x} \mid \mathbf{G}\mathbf{x} \leq \mathbf{b} - \mathbf{A}\mathbf{y}\}\}$, which can be reformulated as $\mathcal{M} = \max\{\mathbf{c}\mathbf{y} + z \mid (\mathbf{y}; z) \in \mathbb{Z}^n \times [l, u]\}$, where $l, u \in \mathbb{R}$ are some valid bounds on z induced from the boundedness of \mathcal{H} . Here, \mathcal{M} is the master problem and z represents the objective value of the subproblem $\max_{\mathbf{x}} \{\mathbf{d}\mathbf{x} \mid \mathbf{G}\mathbf{x} \leq \mathbf{b} - \mathbf{A}\bar{\mathbf{y}}\}$ for any given $\bar{\mathbf{y}}$ as an optimal solution of the master problem. The outcome of the subproblems is either an optimality cut or a feasibility cut that will be added to the master problem. Then, the master problem will be resolved. Proposition 1 implies that formulation \mathcal{M} can be directly modeled and solved with DDs. For this DD, we assign n arc layers to the integer variables y_1, y_2, \dots, y_n , and one arc layer to the continuous variable z with only two arc labels showing a lower and upper bound for this variable. To find an optimal solution, the longest path is calculated, which will be used to solve the subproblems. Note that since \mathcal{M} is a maximization problem, a longest path of the associated DD encodes an optimal solution, and its length gives the optimal value; see Example 2. The feasibility and optimality cuts generated by the subproblems will then be added to *refine* the DD, whose longest path will be recalculated.

The refinement technique consists of removing arcs of the DD that lead to solutions that violate the added inequality, as well as splitting nodes of the DD that lead to different subsequent partial assignments; see Bergman et al. (2016a) for a detailed account on DD refinement techniques. We illustrate this approach in Example 2.

EXAMPLE 2. Suppose that $\max\{2y_1 + 4y_2 + z \mid \mathbf{y} \in \mathcal{P}, z \leq 25\}$ forms the master problem at the penultimate iteration of a BD algorithm, where $\mathcal{P} = \{(0, 0), (1, 1)\}$. This problem is represented by the DD \mathcal{D} in Figure 3(a) where $-M$ is a valid lower bound for z . The longest path of \mathcal{D} encodes the solution $(\hat{y}_1, \hat{y}_2, \hat{z}) = (1, 1, 25)$. Assume that using the point $(\hat{y}_1, \hat{y}_2) = (1, 1)$ in the associated subproblem generates an optimality cut $z \leq 3y_1 + 2y_2 + 10$ for the final iteration of the BD algorithm. Refining DD \mathcal{D} with respect to this cut yields the new DD in Figure 3(b). The longest path represents the optimal solution $(y_1^*, y_2^*, z^*) = (1, 1, 15)$ with length 21, which is the optimal value.

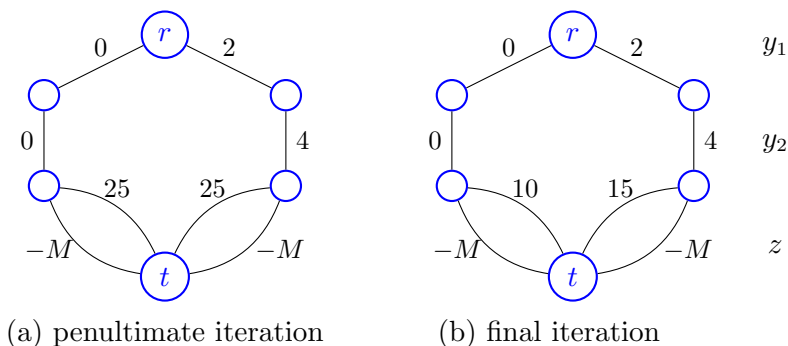


Figure 3 The last two iterations of solving the master problem in Example 2

Using the DD framework as outlined above can be computationally challenging due to exponential growth rate of the size of an exact DD. To mitigate this difficulty, restricted/relaxed DDs can be employed inside of the BD framework as demonstrated in Algorithm 1. We refer to this solution method as DD-BD (Salemi and Davarnia 2022a).

In explaining the steps of Algorithm 1, let point $\hat{\mathbf{y}} \in \mathbb{Z}^k$, where $k \leq n$, be a partial value assignment to the first k coordinates of variable \mathbf{y} , i.e., $y_i = \hat{y}_i$ for all $i \in [k]$. We record the set of all partial value assignments in $\hat{\mathcal{Y}} = \{\hat{\mathbf{y}} \in \mathbb{Z}^k \mid k \in [n]\} \cup \{\ominus\}$, where \ominus represents the case where no coordinate of \mathbf{y} is fixed. Set \mathcal{C} contains the produced Benders cuts throughout the algorithm, and we denote the feasible region described by these cuts by $\mathcal{F}^{\mathcal{C}}$. Further, define $\mathcal{M}^{\mathcal{C}}(\hat{\mathbf{y}}) = \max\{\mathbf{c}\mathbf{y} + z \mid (\mathbf{y}; z) \in \mathbb{Z}^n \times [l, u] \cap \mathcal{F}^{\mathcal{C}}, y_i = \hat{y}_i, \forall i \in [k]\}$ to be the restricted master problem \mathcal{M} obtained through adding cuts in \mathcal{C} and fixing the partial assignment $\hat{\mathbf{y}}$. In this definition, the case with $\mathcal{C} = \emptyset$ and $\hat{\mathcal{Y}} = \{\ominus\}$ is denoted by $\mathcal{M}^0(\ominus) = \mathcal{M}$, which is an input to Algorithm 1.

Algorithm 1: DD-BD**Data:** MIP \mathcal{H} , construction method to build restricted and relaxed DDs for \mathcal{M} **Result:** An optimal solution (\mathbf{y}^*, z^*) and optimal value w^* to \mathcal{H}

```

1 initialize set of partial assignments  $\hat{\mathcal{Y}} = \{\emptyset\}$ , set of Benders cuts  $\mathcal{C} = \emptyset$ , and  $w^* = -\infty$ 
2 if  $\hat{\mathcal{Y}} = \emptyset$  then
3   | terminate and return  $(\mathbf{y}^*, z^*)$  and  $w^*$ 
4 else
5   | select  $\hat{\mathbf{y}} \in \hat{\mathcal{Y}}$  and update  $\hat{\mathcal{Y}} \leftarrow \hat{\mathcal{Y}} \setminus \{\hat{\mathbf{y}}\}$ 
6   | create a restricted DD  $\underline{\mathcal{D}}$  associated with  $\mathcal{M}^C(\hat{\mathbf{y}})$ 
7   | if  $\underline{\mathcal{D}} \neq \emptyset$  then
8     | find a longest  $r$ - $t$  path of  $\underline{\mathcal{D}}$  with encoding point  $(\underline{\mathbf{y}}, \underline{z})$  and length  $\underline{w}$ 
9     | solve the BD subproblem using  $\underline{\mathbf{y}}$  to obtain Benders cut  $\underline{\mathcal{C}}$ 
10    | if  $\underline{\mathcal{C}} \in \mathcal{C}$  then
11      | go to line 17
12    | else
13      | update  $\mathcal{C} \leftarrow \mathcal{C} \cup \underline{\mathcal{C}}$  and refine  $\underline{\mathcal{D}}$  w.r.t.  $\underline{\mathcal{C}}$ 
14      | go to line 8
15    | else
16      | go to line 2
17    | if  $\underline{w} > w^*$  then
18      | update  $w^* \leftarrow \underline{w}$  and  $(\mathbf{y}^*, z^*) \leftarrow (\underline{\mathbf{y}}, \underline{z})$ 
19    | if  $\underline{\mathcal{D}}$  provides an exact representation of  $\mathcal{M}^C(\hat{\mathbf{y}})$  then
20      | go to line 2
21    | else
22      | create a relaxed DD  $\overline{\mathcal{D}}$  associated with  $\mathcal{M}^C(\hat{\mathbf{y}})$ 
23      | find a longest  $r$ - $t$  path of  $\overline{\mathcal{D}}$  with length  $\overline{w}$ 
24      | if  $\overline{w} > w^*$  then
25        | solve the BD subproblem using  $\overline{\mathbf{y}}$  to obtain Benders cut  $\overline{\mathcal{C}}$ 
26        | if  $\overline{\mathcal{C}} \in \mathcal{C}$  then
27          | go to line 31
28        | else
29          | update  $\mathcal{C} \leftarrow \mathcal{C} \cup \overline{\mathcal{C}}$  and refine  $\overline{\mathcal{D}}$  w.r.t.  $\overline{\mathcal{C}}$ 
30          | go to line 23
31        | forall  $u$  in the last exact layer of  $\overline{\mathcal{D}}$  do
32          | update  $\hat{\mathcal{Y}} \leftarrow \hat{\mathcal{Y}} \cup \{\tilde{\mathbf{y}}\}$  where  $\tilde{\mathbf{y}}$  encodes longest  $r$ - $u$  path of  $\overline{\mathcal{D}}$ 
33      | go to line 2

```

The algorithm starts with constructing a restricted DD $\underline{\mathcal{D}}$ corresponding to $\mathcal{M}^C(\hat{\mathbf{y}})$ with empty initial values for C and $\hat{\mathbf{y}}$. We then find a longest r - t path of $\underline{\mathcal{D}}$ encoding solution $(\underline{\mathbf{y}}, \underline{z})$. Next, using $\underline{\mathbf{y}}$, we solve the associated subproblem to obtain a feasibility/optimality cut \underline{C} . We add this cut to C , refine $\underline{\mathcal{D}}$ according to it, and find a new longest r - t path. We repeat these steps until no new feasibility/optimality cut is generated. At this point, the length of a longest r - t path of $\underline{\mathcal{D}}$, denoted by \underline{w} , gives a lower bound to the master problem \mathcal{M} , which is also a valid lower bound to the original problem \mathcal{H} . The value of \underline{w} can be used to update w^* , the optimal value of \mathcal{H} at termination. Next, we create a relaxed DD $\overline{\mathcal{D}}$ corresponding to $\mathcal{M}^C(\hat{\mathbf{y}})$. We find a longest r - t path of $\overline{\mathcal{D}}$ that provides an upper bound \overline{w} to \mathcal{M} . If the upper bound \overline{w} is strictly greater than the current value of w^* , we follow steps similarly to the case for $\underline{\mathcal{D}}$ to iteratively refine $\overline{\mathcal{D}}$ w.r.t. feasibility/optimality cuts through solving the subproblems, until no new cut is generated. Next, we perform a specialized branch-and-bound procedure to improve the bound through expanding merged layers of the DD. To this end, we add all the partial assignments associated with nodes in the last exact layer of $\overline{\mathcal{D}}$ (the last node layer in which no nodes are merged) to the collection $\hat{\mathcal{Y}}$. The nodes corresponding to partial assignments in $\hat{\mathcal{Y}}$ are required to be further explored to check whether or not the value of w^* can be improved. That is, the above process is repeated for every node v with partial assignment in $\hat{\mathcal{Y}}$ as the r - v path is fixed in the new restricted/relaxed DDs. The algorithm terminates when $\hat{\mathcal{Y}}$ becomes empty, at which point w^* is the optimal value.

3. DD-BD Formulation for the SGUFP

In this section, we adapt the DD-BD framework described in Section 2.2 to solve the SGUFP.

3.1. MIP Formulation

We study the MIP formulation of the SGUFP based on that of its deterministic counterpart given in Davarnia et al. (2019). Consider a network $G = (V, A)$ with node set $V := V' \cup \{s, t\}$ and arc set A , where s and t are source and sink nodes, respectively. The source node is connected to all the supply nodes in $S \subseteq V'$, and the sink node is connected to all the demand nodes in $D \subseteq V'$. Figure 4 illustrates the general structure of this network. For a node $q \in V$, let $\delta^-(q) := \{i \in V \mid (i, q) \in A\}$ and $\delta^+(q) := \{j \in V \mid (q, j) \in A\}$ show the set of incoming and outgoing neighbors of q , respectively. Define $\bar{V} \subseteq V'$ as a subset of vertices that must satisfy the NSNM requirement. For each node $q \in \bar{V}$, let binary variable $y_{ij}^q \in \{0, 1\}$ represent whether or not the flow entering node $q \in \bar{V}$ through arc (i, q) leaves node q through arc (q, j) . The first stage of SGUFP determines the matching pairs between incoming and outgoing arcs of unsplittable nodes as follows:

$$\max \quad z \tag{1a}$$

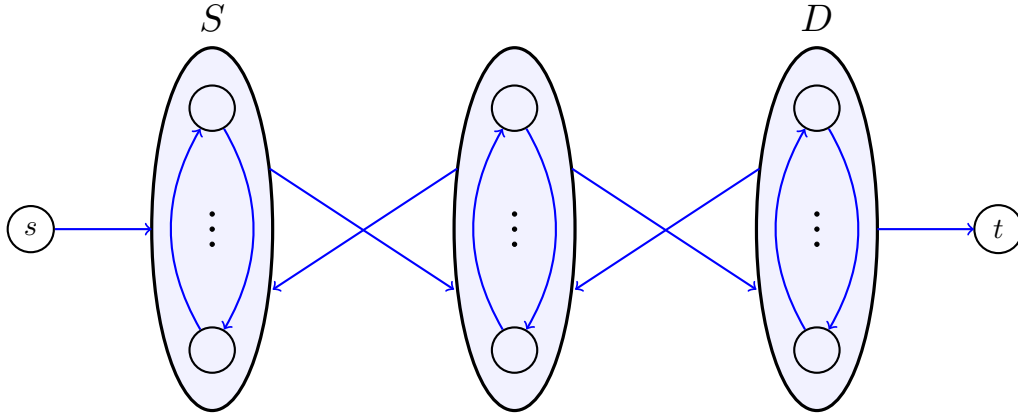


Figure 4 Illustration of network $G = (V' \cup \{s, t\}, A)$

$$\text{s.t.} \quad \sum_{j \in \delta^+(q)} y_{ij}^q \leq 1 \quad \forall i \in \delta^-(q), \forall q \in \bar{V} \quad (1b)$$

$$\sum_{i \in \delta^-(q)} y_{ij}^q \leq 1 \quad \forall j \in \delta^+(q), \forall q \in \bar{V} \quad (1c)$$

$$y_{ij}^q \in \{0, 1\} \quad \forall (i, j) \in \delta^-(q) \times \delta^+(q), \forall q \in \bar{V}, \quad (1d)$$

where constraints (1b) ensure that each incoming arc to a node with NSNM requirement is assigned to at most one outgoing arc, and constraints (1c) guarantee that each outgoing arc from such a node is matched with at most one incoming arc.

In (1a)–(1d), variable z represents the objective value of the second stage of SGUFP where the demand uncertainty is taken into account. This demand uncertainty is modeled by a set Ξ of scenarios for the demand vector \mathbf{d}^ξ with occurrence probability \Pr^ξ for each scenario $\xi \in \Xi$. Let continuous variable $x_{ij}^\xi \in \mathbb{R}_+$ denote the flow from node i to node j through arc (i, j) under scenario $\xi \in \Xi$. We further assign a reward r_{ij} per unit flow to be collected by routing flow through arc (i, j) . It follows that $z = \sum_{\xi \in \Xi} \Pr^\xi z^\xi$, where z^ξ is the objective value of the second stage of SGUFP for each scenario $\xi \in \Xi$. This subproblem is formulated as follows for a given \mathbf{y} vector:

$$\max \quad \sum_{q \in V} \sum_{j \in \delta^+(q)} r_{qj} x_{qj}^\xi \quad (2a)$$

$$\text{s.t.} \quad \sum_{i \in \delta^-(q)} x_{iq}^\xi - \sum_{j \in \delta^+(q)} x_{qj}^\xi = 0 \quad \forall q \in V' \quad (2b)$$

$$\ell_{iq}^\xi \leq x_{iq}^\xi \leq u_{iq}^\xi \quad \forall i \in \delta^-(q), \forall q \in V \quad (2c)$$

$$x_{iq}^\xi - x_{qj}^\xi \leq u_{iq}^\xi (1 - y_{ij}^q) \quad \forall (i, j) \in \delta^-(q) \times \delta^+(q), \forall q \in \bar{V} \quad (2d)$$

$$x_{qj}^\xi - x_{iq}^\xi \leq u_{qj}^\xi (1 - y_{ij}^q) \quad \forall (i, j) \in \delta^-(q) \times \delta^+(q), \forall q \in \bar{V} \quad (2e)$$

$$x_{iq}^\xi \leq u_{iq}^\xi \sum_{j \in \delta^+(q)} y_{ij}^q \quad \forall i \in \delta^-(q), \forall q \in \bar{V} \quad (2f)$$

$$x_{qj}^\xi \leq u_{qj}^\xi \sum_{i \in \delta^-(q)} y_{ij}^q \quad \forall j \in \delta^+(q), \forall q \in \bar{V} \quad (2g)$$

$$x_{ij}^\xi \geq 0 \quad \forall (i, j) \in A. \quad (2h)$$

In the above formulation, the objective function captures the total reward collected by routing flows throughout the network (from the source s to the sink t) to satisfy demands. The flow-balance requirements are represented by (2b). Constraints (2c) bound the flow on each arc from below and above. To impose the demand requirement for each scenario $\xi \in \Xi$, we fix $\ell_{qt}^\xi = u_{qt}^\xi = d_q^\xi$ for all demand nodes $q \in D$ with demand d_q^ξ , and leave the lower and upper bound values unchanged for all other arcs. Constraints (2d)–(2g) model the NSNM requirement for each node $q \in \bar{V}$. In particular, (2d) and (2e) ensure that matching arcs (i, q) and (q, j) have equal flows. Constraints (2f) and (2g) guarantee that an arc without a matching pair does not carry any flow. We note here that the Constraint (2b) is implied by other constraints of the above subproblem under the assumption that \mathbf{y} is feasible to the master problem (1a)–(1d). However, we maintain this constraint in the subproblem because the master formulation in our DD-based approach, as will be described in Section 3.2, may produce a solution that is not feasible to (1a)–(1d). As a result, the addition of the Constraint (2b) will lead to a tighter subproblem formulation.

As discussed in Section 2.2, the first step to use the DD-BD algorithm is to decompose the underlying problem into a master and a subproblem. The above two-stage formulation of the SGUFP is readily amenable to BD since the first stage problem (1a)–(1d) can be considered as the master problem together with some valid lower and upper bounds $-\Gamma$ and Γ on z induced from the boundedness of the MIP formulation. For a given \mathbf{y} value obtained from the master problem and a scenario $\xi \in \Xi$, the second stage problem (2a)–(2h) can be viewed as the desired subproblems. The optimality/feasibility cuts obtained from each scenario-based subproblem are then added to the master problem through aggregation as described in Section 3.3.

3.2. DD-BD: Master Problem Formulation

While the DD-BD Algorithm 1 provides a general solution framework for any bounded MIP, its DD component is problem-specific, i.e., it should be carefully designed based on the specific structure of the underlying problem. In this section, we design such an oracle for the SGUFP that represents the feasible region $\{(1b) - (1d), z \in [-\Gamma, \Gamma]\}$ of the master problem (1a)–(1d). To model this feasible region in the original space of $(\mathbf{y}; z)$ variables, a DD would require $\sum_{q \in \bar{V}} |\delta^-(q)| \times |\delta^+(q)|$ arc layers to represent binary variables \mathbf{y} and one arc layer to encode the continuous variable z . Constructing such a DD, however, would be computationally cumbersome due to the large number of the arc layers. To mitigate this difficulty, we take advantage of the structural flexibility of DDs in representing *irregular* variable types that cannot be used in standard MIP models. One such

variable type is the index set, where arc layers represent indices, rather than domain values. We next show that we can remarkably reduce the number of DD arc layers by reformulating the master problem in a transformed space of variables defined over index sets.

Consider a node $q \in \bar{V}$. In the following, we define mappings that assign an index to each incoming and outgoing arc of q . These mappings enable us to define new variables to reduce the number of DD arc layers. Let $\text{ind}^-(i, q)$ be a one-to-one mapping from incoming arcs (i, q) , for $i \in \delta^-(q)$, to the index set $\{1, 2, \dots, |\delta^-(q)|\}$. Similarly, let $\text{ind}^+(q, j)$ be a one-to-one mapping from outgoing arcs (q, j) , for $j \in \delta^+(q)$, to the index set $\{1, 2, \dots, |\delta^+(q)|\}$. For each incoming arc (i, q) with index $h = \text{ind}^-(i, q)$, we define an integer variable $w_h^q \in \{0, 1, \dots, |\delta^+(q)|\}$ such that $w_h^q = 0$ if this incoming arc is not paired with any outgoing arc, and $w_h^q = k > 0$ if this arc is matched with an outgoing arc (q, j) with index $k = \text{ind}^+(q, j)$.

Next, we give a formulation in the space of \mathbf{w} variables that describes the matching between incoming and outgoing arcs of q for all $q \in \bar{V}$. In the following, $\text{sign}(\cdot)$ represents the sign function that returns 1 if its argument is strictly positive, 0 if the argument is zero, and -1 otherwise. Further, the operator $|\cdot|$, when applied on a set, represents the set size; and when applied on a real number, it represents the absolute value.

PROPOSITION 2. *Formulation*

$$\sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i, q)}^q - \text{ind}^+(q, j) \right| \right) \geq |\delta^-(q)| - 1 \quad \forall j \in \delta^+(q), \forall q \in \bar{V} \quad (3a)$$

$$w_{\text{ind}^-(i, q)}^q \in \{0, 1, \dots, |\delta^+(q)|\} \quad \forall i \in \delta^-(q), \forall q \in \bar{V} \quad (3b)$$

models the matching between incoming and outgoing arcs of nodes $q \in \bar{V}$.

Proof. We show the result for a single node $q \in \bar{V}$. The extension to the multiple node case is straightforward as the matching problem for each node is independent from other nodes. For the direct implication, assume that M^q is a matching between incoming and outgoing arcs of q , with elements of the form (i, j) that represent a matching between the incoming arc (i, q) and the outgoing arc (q, j) . We show that variables \mathbf{w} associated with matching pairs in M^q satisfy constraints (3a) and (3b). It follows from the definition of \mathbf{w} that, for each $(i, j) \in M^q$, we have $w_{\text{ind}^-(i, q)}^q = \text{ind}^+(q, j)$. Also, for any $i \in \delta^-(q)$ that does not have a matching pair in M^q , we have $w_{\text{ind}^-(i, q)}^q = 0$. These value assignments show that \mathbf{w} satisfies (3b) as the image of ind^+ mapping is $\{1, \dots, |\delta^+(q)|\}$. For each $i \in \delta^-(q)$ and $j \in \delta^+(q)$, we have $\left| w_{\text{ind}^-(i, q)}^q - \text{ind}^+(q, j) \right| \geq 0$, with equality holding when $(i, j) \in M^q$. For each $j \in \delta^+(q)$, there are two cases. For the first case, assume that $(i, j) \notin M^q$ for any $i \in \delta^-(q)$. As a result, $\left| w_{\text{ind}^-(i, q)}^q - \text{ind}^+(q, j) \right| > 0$ for all $i \in \delta^-(q)$. Applying the $\text{sign}(\cdot)$ function on these terms yields $\text{sign} \left(\left| w_{\text{ind}^-(i, q)}^q - \text{ind}^+(q, j) \right| \right) = 1$, which implies that

$\sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q,j) \right| \right) = |\delta^-(q)|$, satisfying (3a). For the second case, assume that $(i^*, j) \in M^q$ for some $i^* \in \delta^-(q)$. As a result, we have $\sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q,j) \right| \right) = |\delta^-(q)| - 1$ since $\text{sign} \left(\left| w_{\text{ind}^-(i^*,q)}^q - \text{ind}^+(q,j) \right| \right) = \left| w_{\text{ind}^-(i^*,q)}^q - \text{ind}^+(q,j) \right| = 0$, satisfying (3a).

For the reverse implication, assume that \mathbf{w} is a feasible solution to (3a)–(3b). We show that the pairs of the form (i, j) encoded by these variables constitute a feasible matching between incoming and outgoing arcs of q , i.e., (i) each arc (i, q) is matched with at most one arc (q, j) , and (ii) each arc (q, j) is matched with at most one arc (i, q) . It follows from constraint (3b) that, for each $i \in \delta^-(q)$, variable $w_{\text{ind}^-(i,q)}^q$ takes a value between $\{0, 1, \dots, |\delta^+(q)|\}$. If $w_{\text{ind}^-(i,q)}^q = 0$, then (i, q) is not matched with any outgoing arc, otherwise it is matched with arc (q, j) with $\text{ind}^+(q, j) = w_{\text{ind}^-(i,q)}^q$. This ensures that condition (i) above is satisfied for this matching collection. Further, for each $j \in \delta^+(q)$, constraint (3a) implies that $\text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| \right)$ can be equal to zero for at most one $i \in \delta^-(q)$. In such a case, we would have at most one matching pair of the form (i, j) in the collection, showing that condition (ii) above is satisfied. \square

It follows from Proposition 2 that constraints (3a)–(3b) can replace (1b)–(1d) in the master problem (1a)–(1d) to obtain the following master problem in a transformed space of variables.

$$\max_{\mathbf{w}; z} \{z \mid (3a) - (3b), z \in [-\Gamma, \Gamma]\}. \tag{4}$$

Note that formulation (4) is an integer nonlinear program (INLP) with nonconvex and non-continuous constraint functions. Such a formulation is extremely difficult for conventional MINLP techniques and solvers to handle. However, due to structural flexibility of DDs in representing integer nonlinear programs, this problem can be easily modeled via a DD; see Davarnia and Van Hoeve (2020) for a detailed account on using DDs for modeling INLPs. In the following, we present an algorithm to construct DDs in the space of $(\mathbf{w}; z)$ variables for the master problem (4) with a single node $q \in \bar{V}$. The extension to the case with multiple nodes follows by replicating the DD structure. The output of Algorithm 2 is a DD with $|\delta^-(q)| + 1$ arc layers where the first $|\delta^-(q)|$ layers represent \mathbf{w} variables and the last layer encodes variable z . In this algorithm, s_u denotes the state value of DD node u . The core idea of the algorithm is to use unpaired outgoing arcs of q as the state value at each DD layer that represents the matching for an incoming arc of q .

Next, We show that the solution set of the DD constructed by Algorithm 2 *represents* the feasible region of (4). Note here that DD representation of a MIP set, as described in Section 2.2, does not imply the encoding of all of the solutions of the set, but rather the encoding of a subset of all solutions that subsumes all the extreme points of the set. Such a representation is sufficient to solve an optimization problem over the set with an objective function convex in continuous variables, which is the case for (4).

Algorithm 2: Construction of DD for the master problem of SGUFP with a node $q \in \bar{V}$

Data: node $q \in \bar{V}$, parameter Γ

Result: an exact DD \mathcal{D}

- 1 create the root node $r \in \mathcal{U}_1$ with state $s_r = \{0, 1, \dots, |\delta^+(q)|\}$
 - 2 **forall** $i \in \{1, 2, \dots, |\delta^-(q)|\}$ and $u \in \mathcal{U}_i$ **do**
 - 3 **forall** $\ell \in s_u$ **do**
 - 4 create a node $v \in \mathcal{U}_{i+1}$ with state $(s_u \setminus \{\ell\}) \cup \{0\}$ and an arc $a \in \mathcal{A}_i$ connecting u to v
 with label $l(a) = \ell$
 - 5 **forall** $u \in \mathcal{U}_{1+|\delta^-(q)|}$ **do**
 - 6 create two arcs $a_1, a_2 \in \mathcal{A}_{1+|\delta^-(q)|}$ connecting u to the terminal node with labels $l(a_1) = \Gamma$
 and $l(a_2) = -\Gamma$.
-

THEOREM 1. Consider a SGUFP with $\bar{V} = \{q\}$. Let \mathcal{D} be a DD constructed by Algorithm 2. Then, $\text{Sol}(\mathcal{D})$ represents the feasible region of (4).

Proof. (\subseteq) Consider an r - t path of \mathcal{D} that encodes solution $(\tilde{\mathbf{w}}^q, z)$. According to Algorithm 2, the labels of the first $|\delta^-(q)|$ arcs of this path belong to $\{0, 1, \dots, |\delta^+(q)|\}$, showing that $\tilde{\mathbf{w}}^q$ satisfies constraints (3b). Assume by contradiction that $\tilde{\mathbf{w}}^q$ does not satisfy constraints (3a), i.e., $\sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| \right) \leq |\delta^-(q)| - 2$ for some $j \in \delta^+(q)$. This implies that $\tilde{w}_{\text{ind}^-(i',q)}^q = \tilde{w}_{\text{ind}^-(i'',q)}^q = \text{ind}^+(q, j)$ for two distinct $i', i'' \in \delta^-(q)$. In other words, the arcs at layers $\text{ind}^-(i', q)$ and $\text{ind}^-(i'', q)$ of the selected r - t path both share the same label value $\text{ind}^+(q, j)$. According to line 3 of Algorithm 2, we must have that the state value of nodes at layers $\text{ind}^-(i', q)$ and $\text{ind}^-(i'', q)$ of the r - t path both contain $\text{ind}^+(q, j)$. This is a contradiction to the state update policy in line 4 of Algorithm 2, since positive arc labels at each layer of the DD will be excluded from the state value of the subsequent nodes.

(\supseteq) Consider a feasible solution point $(\tilde{\mathbf{w}}^q; \tilde{z})$ of (4). Suppose $\tilde{\mathbf{w}}^q = (\ell_1, \ell_2, \dots, \ell_{|\delta^-(q)|})$. According to constraints (3a), no two coordinates of $\tilde{\mathbf{w}}^q$ have the same positive value. The state value at the root node in \mathcal{D} contains all index values $\{0, 1, \dots, |\delta^+(q)|\}$. According to Algorithm 2, there exists an arc with label ℓ_1 at the first layer of \mathcal{D} . The state value at the head node of this arc, therefore, contains $\ell_2 \in \{0, 1, \dots, |\delta^+(q)|\} \setminus \{\ell_1\}$, which guarantees an arc with label ℓ_2 at the second layer of this path. Following a similar approach, we can track a path from the root to layer $|\delta^-(q)|$ whose arcs labels match values of $\tilde{\mathbf{w}}^q$. Note for the last layer that $\tilde{z} \in [-\Gamma, \Gamma]$, which is included in the interval between arc labels of the last layer of \mathcal{D} . As a result, $(\tilde{\mathbf{w}}^q; \tilde{z})$ is represented by an r - t path of \mathcal{D} . \square

The main purpose of using a DD that models the master problem (4) over one that models (1a)-(1d) is the size reduction in arc layers that represent variables \mathbf{w} as compared with variables

y. It turns out that this space transformation can significantly improve the solution time of the DD approach. We refer the interested reader to Appendix A for a detailed discussion on these advantages, including preliminary computational results.

Constructing exact DDs as described in Algorithm 2 can be computationally expensive for large size problems. As discussed in Section 2.2, relaxed and restricted DDs are used to circumvent this difficulty. Building restricted DDs is straightforward as it involves the selection of a subset of r - t paths of the exact DD that satisfy a preset width limit. Constructing relaxed DDs, on the other hand, requires careful manipulation of the DD structure to merge nodes in such a way that it encodes a superset of all r - t paths of the exact DD. We demonstrate a method to construct such relaxed DDs in Algorithm 3. Similarly to Algorithm 2, this algorithm is presented for a single NSNM node, but can be extended to multiple nodes by replicating the procedure.

Algorithm 3: Construction of relaxed DD for the master problem of SGUFP with a node $q \in \bar{V}$

Data: node $q \in \bar{V}$, parameter Γ

Result: a relaxed DD $\bar{\mathcal{D}}$

- 1 create the root node $r \in \mathcal{U}_1$ with state $s_r = \{0, 1, \dots, |\delta^+(q)|\}$
 - 2 **forall** $i \in \{1, 2, \dots, |\delta^-(q)|\}$ and $u \in \mathcal{U}_i$ **do**
 - 3 **forall** $\ell \in s_u$ **do**
 - 4 create a node $v \in \mathcal{U}_{i+1}$ with state $(s_u \setminus \{\ell\}) \cup \{0\}$ and an arc $a \in \mathcal{A}_i$ connecting u to v
 with label $l(a) = \ell$
 - 5 select a subset of nodes $v_1, v_2, \dots, v_k \in \mathcal{U}_{i+1}$ and merge them into node v' with state
 $s_{v'} = \bigcup_{j=1}^k s_{v_j}$
 - 6 **forall** $u \in \mathcal{U}_{1+|\delta^-(q)|}$ **do**
 - 7 create two arcs $a_1, a_2 \in \mathcal{A}_{1+|\delta^-(q)|}$ connecting u to the terminal node with labels $l(a_1) = \Gamma$
 and $l(a_2) = -\Gamma$.
-

THEOREM 2. Consider a SGUFP with $\bar{V} = \{q\}$. Let $\bar{\mathcal{D}}$ be a DD constructed by Algorithm 3. Then, $\bar{\mathcal{D}}$ represents a relaxation of the feasible region of (4).

Proof. Let $\dot{\mathcal{D}}$ be the DD constructed by Algorithm 2 for the master problem (4) with a single node $q \in \bar{V}$. It suffices to show that the solution set of $\bar{\mathcal{D}}$ provides a relaxation for that of $\dot{\mathcal{D}}$. Pick a root-terminal path \dot{P} of $\dot{\mathcal{D}}$ with encoding point $(\dot{\mathbf{w}}^q; \dot{z})$. We show that there exist a root-terminal path \bar{P} of $\bar{\mathcal{D}}$ with encoding point $(\bar{\mathbf{w}}^q; \bar{z})$ such that $\bar{\mathbf{w}}^q = \dot{\mathbf{w}}^q$ and $\bar{z} = \dot{z}$. Given a DD, define P_k to be a sub-path composed of arcs in the first k layers, for $1 \leq k \leq |\delta^-(q)|$. We show for any sub-path \dot{P}_k of $\dot{\mathcal{D}}$ with encoding point $\dot{\mathbf{w}}_k^q = (\dot{w}_1^q, \dots, \dot{w}_k^q)$, there exists a sub-path \bar{P}_k of $\bar{\mathcal{D}}$ with encoding

point $\bar{\mathbf{w}}_k = (\bar{w}_1, \dots, \bar{w}_k)$ such that $\bar{\mathbf{w}}_h = \dot{\mathbf{w}}_h$ for $h = 1, \dots, k$. Note that we only need to prove the matching values for $k \leq |\delta^-(q)|$, because each node at node layer $|\delta^-(q)| + 1$ of both $\dot{\mathcal{D}}$ and $\bar{\mathcal{D}}$ is connected by two arcs with labels $-\Gamma$ and Γ to the terminal node, and thus there are always matching arcs with the same label for the last layer, i.e., $\bar{z} = \dot{z}$. We prove the result by induction on k . The base case for $k = 1$ is trivial, since $\bar{\mathcal{D}}$ contains arcs with labels $\{0, 1, \dots, |\delta^+(q)|\}$ in the first layer, which includes the label value of the first arc on \dot{P}_1 . For the induction hypothesis, assume that the statement is true for $k = d$, i.e., for the sub-path \dot{P}_d with label values $\dot{\mathbf{w}}_d^q = (\dot{w}_1^q, \dots, \dot{w}_d^q)$, there is sub-path \bar{P}_d of $\bar{\mathcal{D}}$ with matching arc labels. We show the statement holds for $d + 1$. Let $u \in \dot{A}_{d+1}$ and $v \in \bar{A}_{d+1}$ be the end nodes of \dot{P}_d and \bar{P}_d , respectively. It follows from Algorithm 2 that the index set representing the state value at node u contains \dot{w}_{d+1}^q , i.e., $\dot{w}_{d+1}^q \in \dot{s}_u = \{0\} \cup \{1, \dots, |\delta^+(q)|\} \setminus \{\dot{w}_1, \dot{w}_2, \dots, \dot{w}_d\}$. The merging step in line 5 of Algorithm 3, on the other hand, implies that $\bar{s}_v \supseteq \{0\} \cup \{1, \dots, |\delta^+(q)|\} \setminus \{\bar{w}_1, \bar{w}_2, \dots, \bar{w}_d\} = \{0\} \cup \{1, \dots, |\delta^+(q)|\} \setminus \{\dot{w}_1, \dot{w}_2, \dots, \dot{w}_d\} = \dot{s}_u$, where the inclusion follows from the fact that state values at nodes on path \bar{P}_d contain those of each individual path due to merging operation, and the first equality holds because of the induction hypothesis. As a result, \bar{s}_v must contain \dot{w}_{d+1}^q , which implies that there exists an arc with \dot{w}_{d+1}^q connected to node v on \bar{P}_d . Attaching this arc to \bar{P}_d , we obtain the desired sub-path \bar{P}_{d+1} . \square

3.3. DD-BD: Subproblem Formulation

At each iteration of the DD-BD algorithm, an optimal solution of the master problem is plugged into the subproblems to obtain feasibility/optimality cuts. For the SGUFP formulation, this procedure translates to obtaining an optimal solution of (4) in the space of \mathbf{w} variables, which is used to solve the subproblem (2a)-(2h). The formulation of the subproblem, however, is defined over the original binary variables \mathbf{y} , and the resulting feasibility/optimality cuts are generated in this space. To remedy this discrepancy between the space of variables in the master and subproblems, we need to find a one-to-one mapping between variables \mathbf{w} and \mathbf{y} , as outlined next.

PROPOSITION 3. *Consider a node $q \in \bar{V}$. Let \mathbf{y}^q be a feasible solution to (1b)-(1d). Then, \mathbf{w}^q obtained as*

$$w_{\text{ind}^-(i,q)}^q = \sum_{j \in \delta^+(q)} \text{ind}^+(q, j) y_{ij}^q, \quad \forall i \in \delta^-(q), \quad (5)$$

is a feasible solution to (3a)-(3b). Conversely, let \mathbf{w}^q be a feasible solution to (3a)-(3b). Then, \mathbf{y}^q obtained as

$$y_{ij}^q = 1 - \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| \right) \quad \forall (i, j) \in \delta^-(q) \times \delta^+(q), \quad (6)$$

is a feasible solution to (1b)-(1d).

Proof. For the direct statement, let \mathbf{y}^q be a feasible solution to (1b)-(1d), and construct a vector \mathbf{w}^q according to (5). We show that \mathbf{w}^q satisfies all constraints (3a)-(3b). First, we show that constraints (3a) are satisfied. Assume by contradiction that there exists $j' \in \delta^+(q)$ such that $\sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j') \right| \right) \leq |\delta^-(q)| - 2$. This implies that $w_{\text{ind}^-(i',q)}^q = w_{\text{ind}^-(i'',q)}^q = \text{ind}^+(q, j')$ for some $i', i'' \in \delta^-(q)$. Then, we can write that

$$w_{\text{ind}^-(i',q)}^q = \sum_{j \in \delta^+(q)} \text{ind}^+(q, j) y_{i'j}^q = \text{ind}^+(q, j') = \sum_{j \in \delta^+(q)} \text{ind}^+(q, j) y_{i''j}^q = w_{\text{ind}^-(i'',q)}^q,$$

where the first and last equalities hold by (5). The second and third equalities in the above chain of relations imply that $y_{i'j'}^q = y_{i''j'}^q = 1$, since $\text{ind}^+(q, j') > 0$. This violates constraints (1c), reaching a contradiction. Next, we show that constraints (3b) are satisfied. The proof follows directly from construction of \mathbf{w}^q and constraints (1b).

For the converse statement, let \mathbf{w}^q be a feasible solution to (3a)-(3b), and construct a vector \mathbf{y}^q according to (6). We show that \mathbf{y}^q satisfies all constraints (1b)-(1d). To show that each constraint (1b) is satisfied, consider $i \in \delta^-(q)$. We can write that

$$\sum_{j \in \delta^+(q)} y_{ij}^q = |\delta^+(q)| - \sum_{j \in \delta^+(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| \right) \leq |\delta^+(q)| - (|\delta^+(q)| - 1) = 1,$$

where the first equality follows from the construction of \mathbf{y}^q , and the inequality holds by (3b) as $\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| = 0$ for at most one index $j \in \delta^+(q)$. To show that each constraint (1c) is satisfied, select $j \in \delta^+(q)$. We have

$$\sum_{i \in \delta^-(q)} y_{ij}^q = |\delta^-(q)| - \sum_{i \in \delta^-(q)} \text{sign} \left(\left| w_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j) \right| \right) \leq 1,$$

where the equality follows from the construction of \mathbf{y}^q , and the inequality holds because of constraint (3a). Finally, each constraint (1d) is satisfied due to the fact that $1 - \text{sign}(|\cdot|) \in \{0, 1\}$.

□

PROPOSITION 4. *Mappings described by (5) and (6) are one-to-one over their respective domains.*

Proof. Note that the mapping described by (5) is a linear transformation of the form $\mathbf{w}^q = B\mathbf{y}^q$ with coefficient matrix $B \in \mathbb{Z}^{|\delta^-(q)| \times (|\delta^-(q)| + |\delta^+(q)|)}$. It is clear from the identity block structure of B , that it is full row-rank, since each column contains a single non-zero element while each row has at least one non-zero element. As a result, the null space of B is the origin, which implies that $\hat{\mathbf{w}}^q = \tilde{\mathbf{w}}^q$ only if $\hat{\mathbf{y}}^q = \tilde{\mathbf{y}}^q$.

For the mapping described by (6), let distinct points $\hat{\mathbf{w}}^q$ and $\tilde{\mathbf{w}}^q$ satisfy (3b). Construct vectors $\hat{\mathbf{y}}^q$ and $\tilde{\mathbf{y}}^q$ by (6) using $\hat{\mathbf{w}}^q$ and $\tilde{\mathbf{w}}^q$, respectively. Because $\hat{\mathbf{w}}^q$ and $\tilde{\mathbf{w}}^q$ are distinct, there must

exist $i \in \delta^-(q)$ such that $\hat{w}_{\text{ind}^-(i,q)}^q \neq \tilde{w}_{\text{ind}^-(i,q)}^q$. This implies that at least one of these variables, say $\hat{w}_{\text{ind}^-(i,q)}^q$, is non-zero. It follows from (3b) that $\hat{w}_{\text{ind}^-(i,q)}^q = \text{ind}^+(q, j')$ for some $j' \in \delta^+(q)$, and that $\hat{w}_{\text{ind}^-(i,q)}^q \neq \text{ind}^+(q, j')$. According to (6), we write that $\hat{y}_{ij'} = 1 - \text{sign}\left(\left|\hat{w}_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j')\right|\right) = 1$, and that $\tilde{y}_{ij'} = 1 - \text{sign}\left(\left|\tilde{w}_{\text{ind}^-(i,q)}^q - \text{ind}^+(q, j')\right|\right) = 0$, showing that $\hat{\mathbf{y}}^q \neq \tilde{\mathbf{y}}^q$. \square

Using the results of Propositions 3 and 4, we can apply the DD-BD Algorithm 1 in its entirety for the SGUFP. In particular, at each iteration of the algorithm, we can transform the optimal solution $(\bar{\mathbf{w}}, \bar{z})$ obtained from the DD representing the master problem (4) into a solution $(\bar{\mathbf{y}}, \bar{z})$ through the mapping (6). Given an optimal first-stage solution $\bar{\mathbf{y}}$, we can solve $|\Xi|$ separate subproblems; one for each demand realization in the second-stage. The feasibility cuts obtained from subproblems, which are in the space of \mathbf{y} variables, are translated back into the space of \mathbf{w} variables through the mapping (5) and added to the master problem. Further, in a case where all subproblems produce an optimality cut, they can be aggregated to generate an optimality cut in the space of (\mathbf{y}, z) , which is added to the master problem after being translated into the space of (\mathbf{w}, z) variables. The master DD will be refined with respect to the resulting inequalities, and an optimal solution is returned to be used for the next iteration.

In the remainder of this section, we present details on the derivation of optimality/feasibility cuts from subproblem (2a)-(2h). Consider the following partitioning of the set of arcs A into subsets

$$\begin{aligned} A_1 &:= \{(i, j) \in A \mid \delta^-(i) = \emptyset, \delta^+(j) \neq \emptyset\}, \quad A_2 := \{(i, j) \in A \mid \delta^-(i) \neq \emptyset, \delta^+(j) = \emptyset\}, \\ A_3 &:= \{(i, j) \in A \mid \delta^-(i) \neq \emptyset, \delta^+(j) \neq \emptyset\}, \quad A_4 := \{(i, j) \in A \mid \delta^-(i) = \emptyset, \delta^+(j) = \emptyset\}, \end{aligned}$$

and let $\boldsymbol{\theta}^\xi = (\beta^\xi, \gamma^\xi, \boldsymbol{\delta}^\xi, \boldsymbol{\phi}^\xi, \boldsymbol{\lambda}^\xi, \boldsymbol{\mu}^\xi)$ be the vector of dual variables associated with constraints of (2a)-(2h) for a scenario $\xi \in \Xi$. Further, define the bi-function

$$\begin{aligned} f(\mathbf{y}; \boldsymbol{\theta}^\xi) &= \sum_{q \in V} \sum_{j \in \delta^+(q)} (-\ell_{qj} \beta_{qj}^\xi + u_{qj} \gamma_{qj}^\xi) + \sum_{q \in \bar{V}} \sum_{(i,j) \in \delta^-(q) \times \delta^+(q)} (u_{iq}(1 - y_{ij}^q) \lambda_{iqj}^\xi + u_{qj}(1 - y_{ij}^q) \mu_{iqj}^\xi) \\ &+ \sum_{q \in \bar{V}} \sum_{i \in \delta^-(q)} \left(u_{iq} \sum_{j \in \delta^+(q)} y_{ij}^q \sigma_{iq}^\xi \right) + \sum_{q \in \bar{V}} \sum_{j \in \delta^+(q)} \left(u_{qj} \sum_{i \in \delta^-(q)} y_{ij}^q \phi_{qj}^\xi \right). \end{aligned}$$

For a given $\bar{\mathbf{y}}$ and each scenario $\xi \in \Xi$, the dual of the subproblem (2a)-(2h) can be written as follows where the symbol \star on a node means that it belongs to \bar{V} .

$$\min f(\bar{\mathbf{y}}; \boldsymbol{\theta}^\xi) \tag{7a}$$

$$\text{s.t. } \alpha_q^\xi - \beta_{i_q}^\xi + \gamma_{i_q}^\xi + \sum_{j: j \in \delta^+(\hat{q})} \lambda_{i_q j}^\xi - \sum_{j: j \in \delta^+(\hat{q})} \mu_{i_q j}^\xi + \sigma_{i_q}^\xi \geq r_{i_q} \quad \forall (i, \hat{q}) \in A_1 \tag{7b}$$

$$\alpha_q^\xi - \beta_{i_q}^\xi + \gamma_{i_q}^\xi \geq r_{i_q} \quad \forall (i, q) \in A_1 \tag{7c}$$

$$-\alpha_q^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi - \sum_{i:i \in \delta^-(\hat{q})} \lambda_{iqj}^\xi + \sum_{i:i \in \delta^-(\hat{q})} \mu_{iqj}^\xi + \phi_{qj}^\xi \geq r_{qj}^* \quad \forall(\hat{q}, j) \in A_2 \quad (7d)$$

$$-\alpha_q^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi \geq r_{qj} \quad \forall(q, j) \in A_2 \quad (7e)$$

$$-\alpha_q^\xi + \alpha_j^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi + \sum_{i \in \delta^-(\hat{q})} (\mu_{iqj}^\xi - \lambda_{iqj}^\xi) + \sum_{i \in \delta^+(\hat{j})} (\lambda_{qji}^\xi - \mu_{qji}^\xi) + \sigma_{qj}^\xi + \phi_{qj}^\xi \geq r_{qj}^* \quad \forall(\hat{q}, \hat{j}) \in A_3 \quad (7f)$$

$$-\alpha_q^\xi + \alpha_j^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi + \sum_{i \in \delta^-(\hat{q})} (\mu_{iqj}^\xi - \lambda_{iqj}^\xi) + \phi_{qj}^\xi \geq r_{qj} \quad \forall(\hat{q}, j) \in A_3 \quad (7g)$$

$$-\alpha_q^\xi + \alpha_j^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi + \sum_{i \in \delta^+(\hat{j})} (\lambda_{qji}^\xi - \mu_{qji}^\xi) + \sigma_{qj}^\xi \geq r_{qj}^* \quad \forall(q, \hat{j}) \in A_3 \quad (7h)$$

$$-\alpha_q^\xi + \alpha_j^\xi - \beta_{qj}^\xi + \gamma_{qj}^\xi \geq r_{qj} \quad \forall(q, j) \in A_3 \quad (7i)$$

$$-\beta_{iq}^\xi + \gamma_{iq}^\xi \geq r_{iq} \quad \forall(i, q) \in A_4 \quad (7j)$$

$$\alpha_q^\xi \in \mathbb{R} \quad \forall q \in V' \quad (7k)$$

$$\beta_{ij}^\xi, \gamma_{ij}^\xi, \sigma_{ij}^\xi, \phi_{ij}^\xi, \lambda_{iqj}^\xi, \mu_{iqj}^\xi \geq 0 \quad \forall i, q, j \in V. \quad (7l)$$

If the above problem has an optimal solution $\hat{\theta}^\xi$ for all $\xi \in \Xi$, the output of the subproblems will be an optimality cut of the form $\sum_{\xi \in \Xi} \Pr^\xi f(\mathbf{y}; \hat{\theta}^\xi) \geq z$. If the above problem is unbounded along a ray $\hat{\theta}^\xi$ for a $\xi \in \Xi$, the output of the subproblem will be a feasibility cut of the form $f(\mathbf{y}; \hat{\theta}^\xi) \geq 0$. Note that replacing variables \mathbf{y} in the above constraints with \mathbf{w} through the mapping (5) results in separable nonlinear constraints. Nevertheless, since these constraints will be used to refine the master DD, their incorporation is simple due to structural flexibility of DDs in modeling such constraints; we refer the reader to Davarnia and Van Hoeve (2020) for a detailed account for modeling INLPs with DDs.

4. Computational Experiments

In this section, we solve SGUFP as a core model for the unit train scheduling problem with demand stochasticity using three different approaches: (i) the standard MIP formulation that is a deterministic equivalent of the two-stage model and contains all variables and constraints of the master problem and $|\Xi|$ subproblems; (ii) the Benders reformulation presented in Section 3.1 composed of the master problem (1a)-(1d) and $|\Xi|$ subproblems (2a)-(2h); and (iii) the DD-BD algorithm proposed in the present paper. In the Benders approach, we solve separate subproblems using a fixed vector $\bar{\mathbf{y}}$ obtained from the master problem. The feasibility cuts generated by subproblems are added directly to the constraint set of the master problem, and the optimality cuts are added as an aggregated cut over all scenarios. We note here that when there is a feasibility cut for any scenario, we add it directly to separate the solution of the current iteration and move on to the next iteration. To obtain a valid inequality that provides a bound for the single z variable, we need to aggregate valid inequalities over all scenario subproblems as z is composed of the objective value

of all these subproblems. Therefore, we can only produce an optimality cut for the z variable when we have optimality cuts for all of the subproblems. For the DD-BD approach, we use the following algorithmic choices to build restricted and relaxed DDs. For the restricted DDs, we choose a subset of the r - t paths with largest lengths, which are more likely to contain an optimal solution. For the relaxed DDs, we merge nodes that have the largest number of common members in their state values. We refer the reader to Bergman et al. (2016a) for other heuristic approaches that could be used for this purpose.

4.1. Test Instances

In our experiments, we consider the structure of the SGUFP network given in Section 3.1. To ensure that the problem is always feasible, we create an artificial node s_0 to compensate for any shortage of the supply, and add an arc from the artificial supply s_0 to each demand node.

We create test instances based on the specification given in Davarnia et al. (2019), which is inspired by realistic models. In particular, we consider a base rail network $G' = (V', A')$ where 10% and 30% of the nodes are supply and demand nodes, respectively. We assume that 50% of the nodes must satisfy the NSNM requirement. We then create a network $G = (V, A)$ by augmenting supply/demand and artificial nodes as described above with the following settings. The integer supply value at supply nodes is randomly selected from the interval $[100, 600]$. The capacity of arcs connecting s_0 to demand nodes are considered to be unbounded, and the integer capacity value of other arcs is randomly selected from the interval $[100, 300]$. For each demand scenario $\xi \in \Xi$, the integer demand value at demand nodes is randomly chosen from the interval $[100, 200]$. The reward of the arcs connecting s_0 to the demand nodes are generated from the interval $[-10, -5]$ to represent the cost of lost demands. The reward of the arcs connecting the source to the supply nodes is randomly selected from the interval $[5, 10]$, and the reward of the arcs connecting the demand nodes to the sink is fixed to zero since the flow of these arcs is also fixed. The reward of all other arcs is created randomly from the interval $[-2, 2]$ where the negative values indicate the cost of sending flows through congested arcs. We consider four categories of rail networks with $|V'| \in \{40, 60, 80, 100\}$. For each category, we create five scenario classes for the number of demand scenarios $|\Xi| \in \{50, 100, 150, 200, 250\}$. For each network category and scenario class, we create five random instances based on the above settings. Test instances are publicly available (Salemi and Davarnia 2022b).

4.2. Numerical Results

In this section, we present the numerical results that compare the performance of the DD-BD formulation for the SGUFP instances with that of the MIP formulation, denoted by “MIP”, and the standard Benders reformulation, denoted by “BD”. All experiments are conducted on a machine

running Windows 10, x64 operating system with Intel[®] Core i7 processor (2.60 GHz) and 32 GB RAM. The Gurobi optimization solver (version 9.1.1) is used to solve instances for the MIP and BD models. When solving problems with Gurobi, we turn off presolve and cuts for all methods to have a fair comparison. Tables 1-4 report the running times of each of these formulations for $|V'| \in \{40, 60, 80, 100\}$ and $|\Xi| \in \{50, 100, 150, 200, 250\}$ where the time limit is set to 3600 seconds. The symbol “> 3600” indicates that the problem was not solved within the time limit. As evident in these tables, the DD-BD formulation outperforms the other alternatives. In particular, the gap between the solution time of the DD-BD and the MIP and BD approaches widens as the problem size increases. For example, as reported in Table 1, while the DD-BD approach solves all 25 instances in under 275 seconds, the MIP approach fails to solve 10 of them within 3600 seconds, 80% of which involve 200 or 250 scenarios. This shows a clear superiority of the DD-BD over the MIP method. Further, for most of the instances, the DD-BD approach outperforms the standard BD approach, rendering it as the superior solution method among all three. Figures 5-8 compare the performance of DD-BD, BD, and MIP formulations through box and whisker plots for each network size and under each scenario class. In these figures, for uniformity of illustration, we used 3600 seconds for the running time of instances that fail to solve the problem within that time limit. As the figures show, the minimum, median, and maximum of running times of the DD-BD method are remarkably smaller than those of the both BD and MIP methods in all cases. These results show the potential of the DD-BD framework in solving network problems with challenging combinatorial structures. In Appendix B, we present additional numerical results for the DD-BD approach to assess its ability to solve larger problem sizes.

Table 1 Running times (in seconds) of MIP, BD, and DD-BD for $|V'| = 40$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	MIP	75.74	512.62	2877.19	> 3600	> 3600
	BD	141.83	313.84	339.81	451.93	565.82
	DD-BD	56.94	129.87	163.43	219.02	274.36
2	MIP	67.59	275.07	906.10	1892.21	2235.53
	BD	63.44	121.25	141.04	230.81	235.87
	DD-BD	42.60	82.65	128.16	164.52	208.94
3	MIP	94.86	753.23	2453.05	> 3600	> 3600
	BD	71.14	139.20	172.86	224.33	244.91
	DD-BD	53.32	93.58	113.93	178.65	217.33
4	MIP	71.46	309.62	> 3600	> 3600	> 3600
	BD	63.55	182.01	267.94	334.74	380.22
	DD-BD	46.61	87.81	130.19	183.23	253.72
5	MIP	380.33	406.73	> 3600	> 3600	> 3600
	BD	123.69	198.73	205.16	231.56	287.24
	DD-BD	67.04	104.78	138.46	195.69	231.74

Table 2 Running times (in seconds) of MIP, BD, and DD-BD for $|V'| = 60$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	MIP	893.73	> 3600	> 3600	> 3600	> 3600
	BD	241.85	556.18	582.80	758.54	933.05
	DD-BD	176.16	357.06	603.81	719.27	901.02
2	MIP	206.87	811.64	1554.10	> 3600	> 3600
	BD	259.63	351.39	624.08	816.44	1017.95
	DD-BD	189.07	388.85	572.52	764.76	961.35
3	MIP	139.70	702.96	1035.79	> 3600	> 3600
	BD	246.48	569.37	628.84	795.56	978.15
	DD-BD	142.81	284.65	422.52	565.23	725.86
4	MIP	153.16	415.46	938.03	1681.21	2604.25
	BD	238.33	388.19	563.15	732.59	919.08
	DD-BD	131.29	262.36	393.18	521.12	654.71
5	MIP	165.57	706.16	2447.15	> 3600	> 3600
	BD	194.12	244.61	479.32	463.63	617.09
	DD-BD	112.09	221.30	332.25	443.96	556.33

Table 3 Running times (in seconds) of MIP, BD, and DD-BD for $|V'| = 80$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	MIP	215.82	860.21	> 3600	> 3600	> 3600
	BD	588.51	806.61	1731.50	1860.12	2051.52
	DD-BD	256.12	500.52	757.68	1025.88	1278.13
2	MIP	479.76	> 3600	> 3600	> 3600	> 3600
	BD	398.29	713.01	861.65	1080.79	1709.04
	DD-BD	184.34	379.04	724.66	1088.21	1587.90
3	MIP	238.79	996.22	> 3600	> 3600	> 3600
	BD	702.18	1236.58	1650.42	1773.63	2227.89
	DD-BD	285.13	518.46	778.97	1046.39	1326.22
4	MIP	404.26	2441.64	2855.29	> 3600	> 3600
	BD	572.83	1219.37	1334.21	1745.91	2089.80
	DD-BD	263.78	665.30	1230.81	1277.93	1444.02
5	MIP	778.50	> 3600	> 3600	> 3600	> 3600
	BD	231.11	481.31	625.91	1310.24	1452.27
	DD-BD	187.34	376.96	564.34	1205.54	1412.94

Table 4 Running times (in seconds) of MIP, BD, and DD-BD for $|V'| = 100$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	MIP	774.18	> 3600	> 3600	> 3600	> 3600
	BD	1282.59	1728.71	1848.49	2307.74	3309.93
	DD-BD	698.36	1427.38	1731.95	2014.96	3323.54
2	MIP	480.97	> 3600	> 3600	> 3600	> 3600
	BD	781.47	1573.23	1820.79	2672.18	2819.61
	DD-BD	586.89	1171.96	1848.49	2471.49	2635.22
3	MIP	3071.37	> 3600	> 3600	> 3600	> 3600
	BD	1072.14	1322.96	2112.50	2951.55	3412.99
	DD-BD	485.31	703.70	1055.36	1803.66	2269.97
4	MIP	838.79	2585.38	> 3600	> 3600	> 3600
	BD	1548.93	1738.92	2580.53	2616.19	3169.28
	DD-BD	554.89	743.64	1098.82	2052.73	3094.23
5	MIP	714.39	> 3600	> 3600	> 3600	> 3600
	BD	808.48	1013.68	1722.01	2824.14	3282.10
	DD-BD	353.48	700.57	1680.60	2213.81	2907.78

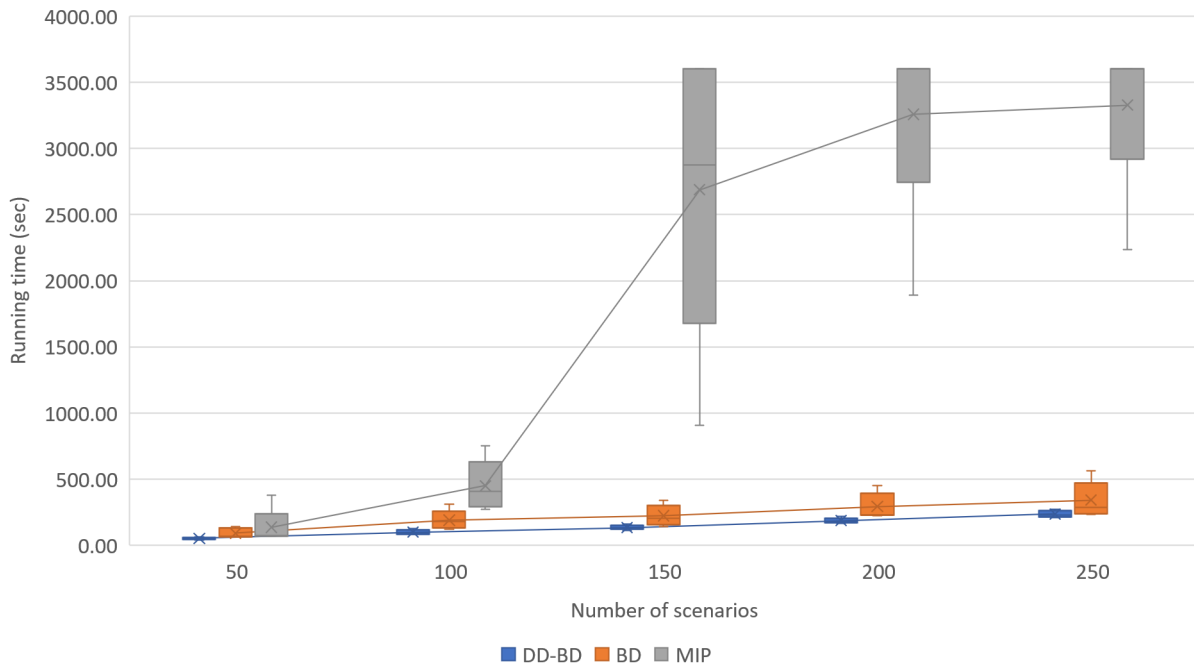


Figure 5 Comparison of DD-BD, BD, and MIP models when $|V'| = 40$ under five scenarios

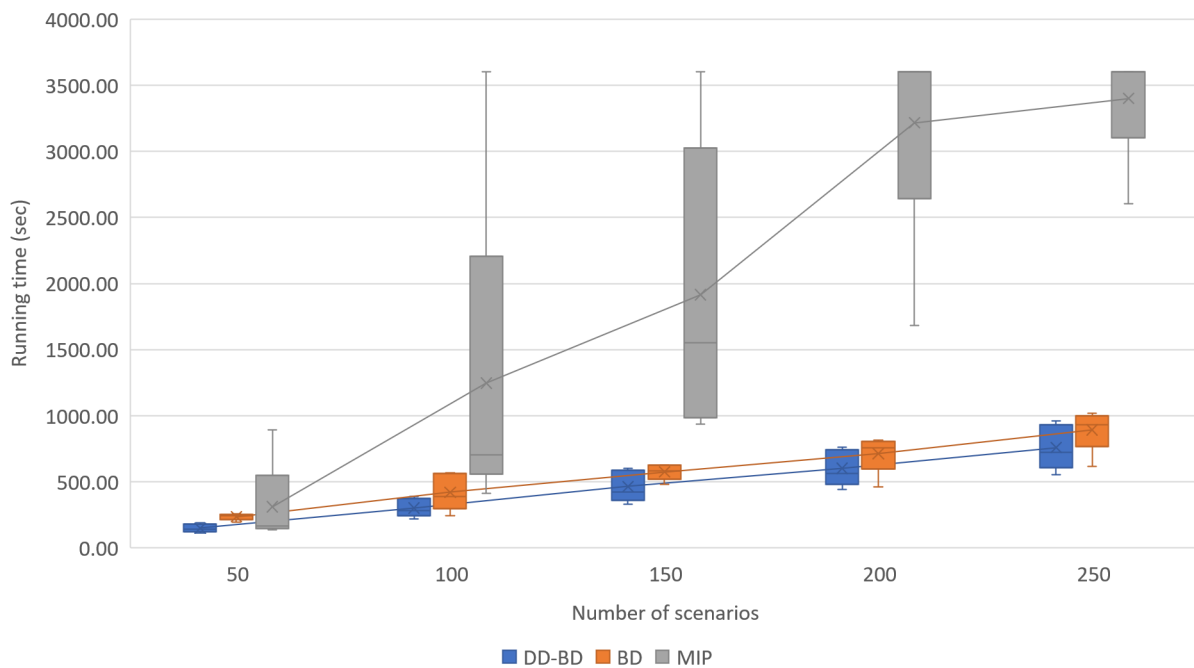


Figure 6 Comparison of DD-BD, BD, and MIP models when $|V'| = 60$ under five scenarios

We conclude this section by noting that, while the focus of this paper has been on the unit train problem with the no-split no-merge requirements, the proposed DD-BD framework can be applied to model network problems that contain additional side constraints on the flow variables, as those constraints can be handled in the subproblems while the DD structure in the master problem

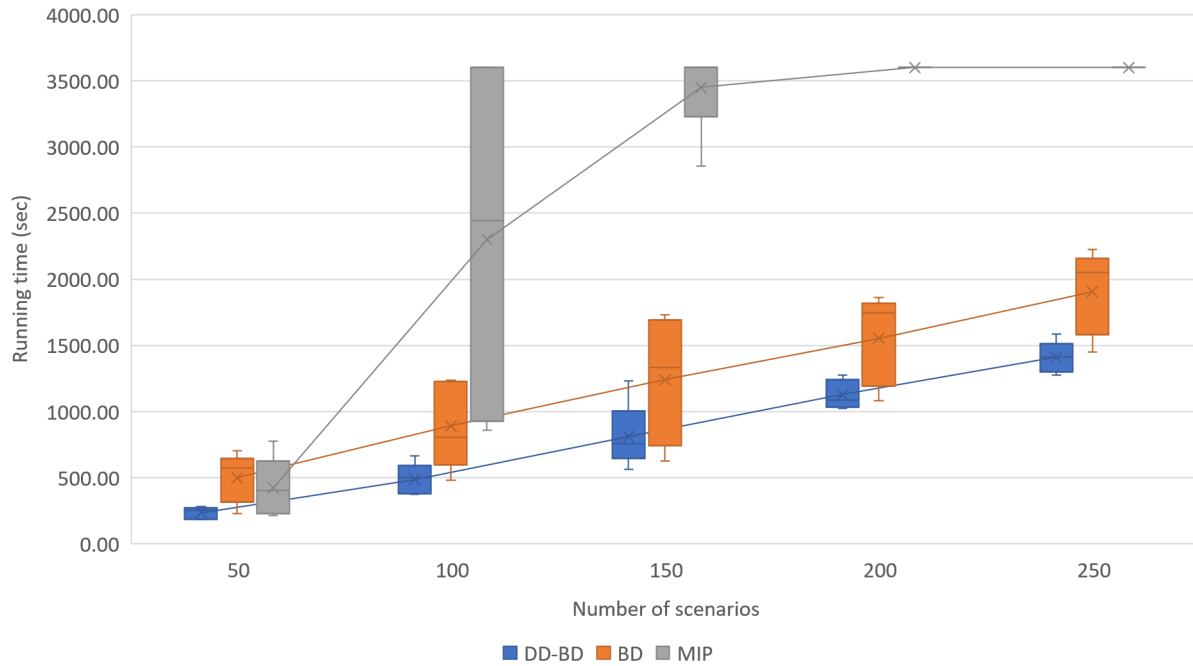


Figure 7 Comparison of DD-BD, BD, and MIP models when $|V'| = 80$ under five scenarios

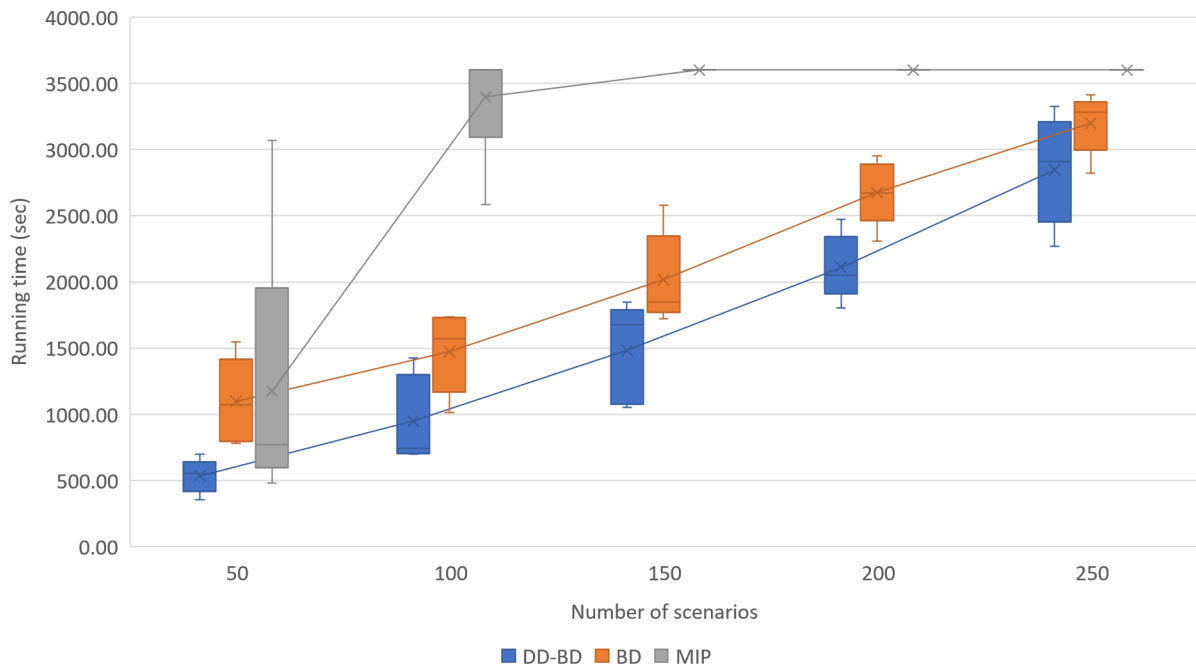


Figure 8 Comparison of DD-BD, BD, and MIP models when $|V'| = 100$ under five scenarios

remains intact. Examples of such side constraints include the *usage-fee* limitation (Holzhauser, Krumke, and Thielen 2017b) and the *flow ratio* requirement (Holzhauser, Krumke, and Thielen 2017a). Applying the DD-BD method to such network models and assessing its effectiveness compared to alternative approaches could be an interesting direction for future research.

5. Conclusion

In this paper, we introduce a DD-based framework to solve the SGUFP. This framework uses Benders decomposition to decompose the SGUFP into a master problem composed of the combinatorial NSNM constraints, and a subproblem that solves a continuous network flow model. The master problem is modeled by a DD, which is successively refined with respect to the cuts generated through subproblems. To assess the performance of the proposed method, we apply it to a variant of unit train scheduling problem formulated as a SGUFP, and compare it with the standard MIP and Benders reformulation of the problem.

Acknowledgments

This project is sponsored in part by the Iowa Energy Center, Iowa Economic Development Authority and its utility partners. We thank the anonymous referees and the Associate Editor for their helpful comments that contributed to improving the paper.

References

- Abbink E, Van den Berg B, Kroon L, Salomon M, 2004 *Allocation of railway rolling stock for passenger trains*. *Transportation Science* 38(1):33–41.
- Alfieri A, Groot R, Kroon L, Schrijver A, 2006 *Efficient circulation of railway rolling stock*. *Transportation Science* 40(3):378–391.
- Andersen HR, Hadzic T, Hooker JN, Tiedemann P, 2007 *A constraint store based on multivalued decision diagrams*. *International Conference on Principles and Practice of Constraint Programming*, 118–132 (Springer).
- Association of American Railroads, 2021 *Freight railroads fact sheet*. <https://www.aar.org>, Accessed: 06/28/2021.
- Baier G, Köhler E, Skutella M, 2005 *The k -splittable flow problem*. *Algorithmica* 42(3):231–248.
- Bergman D, Cire AA, 2018 *Discrete nonlinear optimization by state-space decompositions*. *Management Science* 64(10):4700–4720.
- Bergman D, Cire AA, Van Hoeve WJ, Hooker J, 2016a *Decision diagrams for optimization*, volume 1 (Springer).
- Bergman D, Cire AA, Van Hoeve WJ, Hooker JN, 2016b *Discrete optimization with decision diagrams*. *INFORMS Journal on Computing* 28(1):47–66.
- Borndörfer R, Reuther M, Schlechte T, Waas K, Weider S, 2016 *Integrated optimization of rolling stock rotations for intercity railways*. *Transportation Science* 50(3):863–877.
- Cacchiani V, Toth P, 2012 *Nominal and robust train timetabling problems*. *European Journal of Operational Research* 219(3):727–737.
- Carey M, Crawford I, 2007 *Scheduling trains on a network of busy complex stations*. *Transportation Research Part B: Methodological* 41(2):159–178.
- Ceselli A, Gatto M, Lübbecke ME, Nunkesser M, Schilling H, 2008 *Optimizing the cargo express service of swiss federal railways*. *Transportation Science* 42(4):450–465.
- Chakrabarti A, Chekuri C, Gupta A, Kumar A, 2007 *Approximation algorithms for the unsplittable flow problem*. *Algorithmica* 47(1):53–78.
- Cordeau JF, Toth P, Vigo D, 1998 *A survey of optimization models for train routing and scheduling*. *Transportation Science* 32(4):380–404.
- Cornelsen S, Di Stefano G, 2007 *Track assignment*. *Journal of Discrete Algorithms* 5(2):250–261.
- Davarnia D, 2021 *Strong relaxations for continuous nonlinear programs based on decision diagrams*. *Operations Research Letters* 49(2):239–245.
- Davarnia D, Richard JPP, İċyüz-Ay E, Taslimi B, 2019 *Network models with unsplittable node flows with application to unit train scheduling*. *Operations Research* 67(4):1053–1068.

- Davarnia D, Van Hoeve WJ, 2020 *Outer approximation for integer nonlinear programs via decision diagrams. Mathematical Programming* 1–40.
- Demir E, Burgholzer W, Hrušovský M, Arıkan E, Jammernegg W, Van Woensel T, 2016 *A green inter-modal service network design problem with travel time uncertainty. Transportation Research Part B: Methodological* 93:789–807.
- Fuchsberger M, Lüthi P, 2007 *Solving the train scheduling problem in a main station area via a resource constrained space-time integer multi-commodity flow. Institute for Operations Research ETH Zurich* .
- Furchtgott-Roth D, Hu PS, Nguyen L, Jahanmir S, Moore WH, Riley D, Beningo S, Chambers M, Smith-Pickel S, Thai H, et al., 2021 *Pocket Guide to Transportation 2021* .
- Gong C, Shi J, Wang Y, Zhou H, Yang L, Chen D, Pan H, 2021 *Train timetabling with dynamic and randomness passenger demand: A stochastic optimization method. Transportation Research Part C: Emerging Technologies* 123:102963.
- Gonzalez JE, Cire AA, Lodi A, Rousseau LM, 2020 *Integrated integer programming and decision diagram search tree with an application to the maximum independent set problem. Constraints* 1–24.
- Haahr J, Lusby RM, 2017 *Integrating rolling stock scheduling with train unit shunting. European Journal of Operational Research* 259(2):452–468.
- Haahr JT, Wagenaar JC, Veelenturf LP, Kroon LG, 2016 *A comparison of two exact methods for passenger railway rolling stock (re) scheduling. Transportation Research Part E: Logistics and Transportation Review* 91:15–32.
- Hadžić T, Hooker J, 2006 *Discrete global optimization with binary decision diagrams. Workshop on Global Optimization: Integrating Convexity, Optimization, Logic Programming, and Computational Algebraic Geometry (GICOLAG). Vienna*.
- Harrod S, Gorman MF, 2010 *Operations research for freight train routing and scheduling. Wiley Encyclopedia of Operations Research and Management Science* .
- Heil J, Hoffmann K, Buscher U, 2020 *Railway crew scheduling: Models, methods and applications. European Journal of Operational Research* 283(2):405–425.
- Holzhauser M, Krumke SO, Thielen C, 2017a *Maximum flows in generalized processing networks. Journal of Combinatorial Optimization* 33(4):1226–1256.
- Holzhauser M, Krumke SO, Thielen C, 2017b *A network simplex method for the budget-constrained minimum cost flow problem. European journal of operational research* 259(3):864–872.
- Hosseiniinasab A, Van Hoeve WJ, 2021 *Exact multiple sequence alignment by synchronized decision diagrams. INFORMS Journal on Computing* 33(2):721–738.
- Hu Y, Lan J, Wan C, 2009 *An algorithm for unsplittable flow problem in flexible reconfigurable network. 2009 Fourth International Conference on Frontier of Computer Science and Technology*, 543–547 (IEEE).

- Huntley CL, Brown DE, Sappington DE, Markowicz BP, 1995 *Freight routing and scheduling at CSX transportation*. *Interfaces* 25(3):58–71.
- Içyüz IE, Richard JPP, Eskigun E, Acharya D, 2016 *A two-model solution approach for the monthly coal train reservations planning problem*. *Transportation Science* 50(3):926–946.
- Jin G, He S, Li J, Guo X, Li Y, 2019 *An approach for train stop planning with variable train length and stop time of high-speed rail under stochastic demand*. *IEEE Access* 7:129690–129708.
- Jordan WC, Turnquist MA, 1983 *A stochastic, dynamic network model for railroad car distribution*. *Transportation Science* 17(2):123–145.
- Jovanović D, Harker PT, 1991 *Tactical scheduling of rail operations: the scan i system*. *Transportation Science* 25(1):46–64.
- Kleinberg JM, 1996 *Approximation algorithms for disjoint paths problems*. Ph.D. thesis, Massachusetts Institute of Technology.
- Kolman P, Scheideler C, 2006 *Improved bounds for the unsplittable flow problem*. *Journal of Algorithms* 61(1):20–44.
- Kwan RS, 2011 *Case studies of successful train crew scheduling optimisation*. *Journal of Scheduling* 14(5):423–434.
- Larsen R, Pranzo M, D’Ariano A, Corman F, Pacciarelli D, 2014 *Susceptibility of optimal train schedules to stochastic disturbances of process times*. *Flexible Services and Manufacturing Journal* 26(4):466–489.
- Lawley M, Parmeshwaran V, Richard JP, Turkcan A, Dalal M, Ramcharan D, 2008 *A time-space scheduling model for optimizing recurring bulk railcar deliveries*. *Transportation Research Part B: Methodological* 42(5):438–454.
- Layeb SB, Jaoua A, Jbira A, Makhoulf Y, 2018 *A simulation-optimization approach for scheduling in stochastic freight transportation*. *Computers & Industrial Engineering* 126:99–110.
- Lin Z, Kwan RS, 2014 *A two-phase approach for real-world train unit scheduling*. *Public Transport* 6(1-2):35–65.
- Lin Z, Kwan RS, 2016 *A branch-and-price approach for solving the train unit scheduling problem*. *Transportation Research Part B: Methodological* 94:97–120.
- Lin Z, Kwan RS, 2018 *Redundant coupling/decoupling in train unit scheduling optimization*. *Electronic Notes in Discrete Mathematics* 64:45–54.
- Liu SQ, Kozan E, 2011 *Optimising a coal rail network under capacity constraints*. *Flexible Services and Manufacturing Journal* 23(2):90–110.
- Lusby RM, 2008 *Optimization methods for routing trains through railway junctions*. Ph.D. thesis, ResearchSpace@ Auckland.

- Lusby RM, Larsen J, Ehrgott M, Ryan D, 2011 *Railway track allocation: models and methods*. *OR spectrum* 33(4):843–883.
- Meng L, Zhou X, 2011 *Robust single-track train dispatching model under a dynamic and stochastic environment: A scenario-based rolling horizon solution approach*. *Transportation Research Part B: Methodological* 45(7):1080–1102.
- Quaglietta E, Corman F, Goverde RM, 2013 *Stability of railway dispatching solutions under a stochastic and dynamic environment*. *RailCopenhagen2013: 5th International Seminar on Railway Operations Modelling and Analysis (IAROR)* (Institute for Transport Planning and Systems, ETH Zurich).
- Salemi H, Davarnia D, 2022a *On the structure of decision diagram-representable mixed integer programs with application to unit commitment*. *Operations Research* URL <https://doi.org/10.1287/opre.2022.2353>.
- Salemi H, Davarnia D, 2022b *Test instances for SGUFP*. <https://doi.org/10.5281/zenodo.6373664>.
- Serra T, Hooker JN, 2019 *Compact representation of near-optimal integer programming solutions*. *Mathematical Programming* 1–34.
- Shen Y, Peng K, Chen K, Li J, 2013 *Evolutionary crew scheduling with adaptive chromosomes*. *Transportation Research Part B: Methodological* 56:174–185.
- Sherali HD, Suharko AB, 1998 *A tactical decision support system for empty railcar management*. *Transportation Science* 32(4):306–329.
- Turner C, Tiwari A, Starr A, Blacktop K, 2016 *A review of key planning and scheduling in the rail industry in Europe and UK*. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit* 230(3):984–998.
- Walkowiak K, 2006 *New algorithms for the unsplittable flow problem*. *International Conference on Computational Science and Its Applications*, 1101–1110 (Springer).
- Ying Cs, Chow AH, Chin KS, 2020 *An actor-critic deep reinforcement learning approach for metro train scheduling with rolling stock circulation under stochastic demand*. *Transportation Research Part B: Methodological* 140:210–235.
- Zwaneveld PJ, Kroon LG, Van Hoesel SP, 2001 *Routing trains through a railway station based on a node packing model*. *European Journal of Operational Research* 128(1):14–33.

Appendix A: Comparison of Master Problem Formulations

In this section, we describe the differences between DDs in the space of w variables and those in the space of original y in the master problem formulation (4) in Section 3.2. First, we illustrate the size difference between these DDs in Example 3.

EXAMPLE 3. Consider a directed graph $G = (V, A)$ with node set $V = \{1, 2, q, 3, 4\}$ and arc set $A = \{(1, q), (2, q), (q, 3), (q, 4)\}$ where the central node q is subject to NSNM constraints. Let $\text{ind}^-(1, q) = \text{ind}^+(q, 3) = 1$ and $\text{ind}^-(2, q) = \text{ind}^+(q, 4) = 2$. Then, the exact DDs showed in Figures 9(a) and 9(b) with three and five arc layers represent the feasible region of master problem (4) and (1a)-(1d), respectively, where $-M$ and M are valid bounds for variable z .

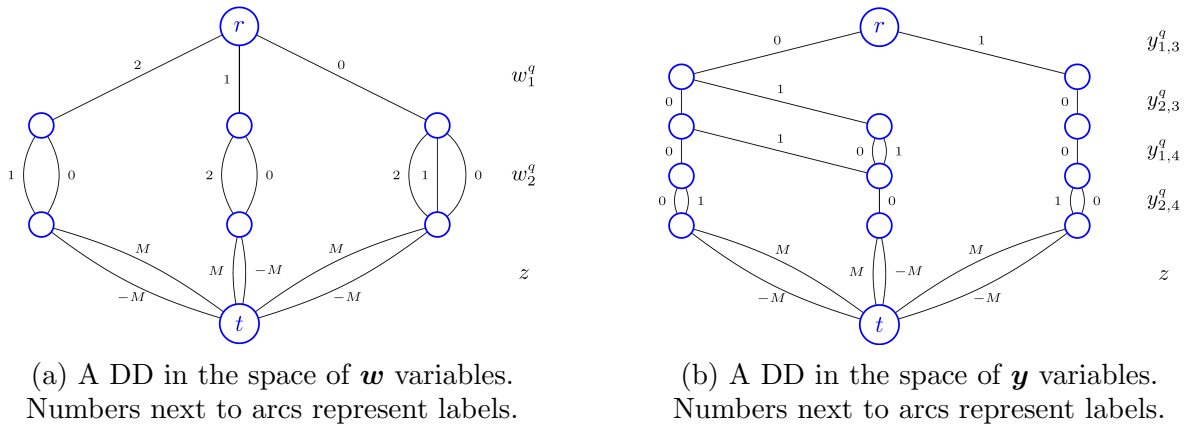


Figure 9 Comparison of the number of arc layers for DDs in the space of w and y variables

As evident from the above example, the main advantage of using a DD in the space of w is the reduction in the number of arc layers, which is the main determinant of the DDs computational efficiency. In particular, even though such a DD has a larger number of nodes at the layers, a relaxed DD can be constructed to limit the width, and hence provide an efficient relaxed DD in a smaller dimension, whereas the relaxations of the DD constructed in the space of y variables would still be higher-dimensional.

To assess the computational efficiency of the solution approach in relation to the DD space, we compare the performance of the DD-BD method under two different settings: (i) where DDs are built in the space of w variables, denoted by DD-BD- w , and (ii) where DDs are built in the space of y variables, denoted by DD-BD- y . We report the results of these two implementations for $|V'| \in \{40, 80\}$ and under five different scenarios in Table 5 and Table 6.

As observed in these tables, the DD-BD- w solves all instances faster than DD-BD- y , with orders of magnitude time improvement as the problem size (number of scenarios) increases. These preliminary computational results show the advantage of designing the DD-BD method for the SGUFP in a transformed space of variables.

Table 5 Running times (in seconds) of DD-BD- w and DD-BD- y for $|V'| = 40$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	DD-BD- w	56.94	129.87	163.43	219.02	274.36
	DD-BD- y	89.68	304.08	432.34	642.70	839.57
2	DD-BD- w	42.60	82.65	128.16	164.52	208.94
	DD-BD- y	68.23	148.76	244.53	344.86	605.04
3	DD-BD- w	53.32	93.58	113.93	178.65	217.33
	DD-BD- y	83.05	157.67	310.07	541.33	658.98
4	DD-BD- w	46.61	87.81	130.19	183.23	253.72
	DD-BD- y	78.11	149.26	325.31	460.73	694.57
5	DD-BD- w	67.04	104.78	138.46	195.69	231.74
	DD-BD- y	109.61	223.78	351.80	532.12	669.78

Table 6 Running times (in seconds) of DD-BD- w and DD-BD- y for $|V'| = 80$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	DD-BD- w	256.12	500.52	757.68	1025.88	1278.13
	DD-BD- y	483.42	977.03	1642.27	3175.72	4230.29
2	DD-BD- w	184.34	379.04	724.66	1088.21	1587.90
	DD-BD- y	340.13	864.21	1856.96	3010.55	4843.67
3	DD-BD- w	285.13	518.46	778.97	1046.39	1326.22
	DD-BD- y	568.32	1176.44	2401.98	3326.76	4283.58
4	DD-BD- w	263.78	665.30	1230.81	1277.93	1444.02
	DD-BD- y	501.04	1430.77	2868.92	3356.39	4356.48
5	DD-BD- w	187.34	376.96	564.34	1205.54	1412.94
	DD-BD- y	354.37	781.18	1279.73	3001.72	3834.08

Appendix B: Additional Computational Experiments

In this section, we present additional numerical results to assess the limits of the DD-BD method for larger problem instances. These results are given in Tables 7 and 8, where the columns are defined similarly to those of Tables 1-4. For these instances, the time limit is set to 3600 seconds, and the symbol “> 3600” indicates that the problem is not solved within this time limit.

Table 7 Running times (in seconds) of DD-BD for $|V'| = 120$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	DD-BD	1494.49	2824.58	> 3600	> 3600	> 3600
2	DD-BD	975.47	1892.41	3198.18	> 3600	> 3600
3	DD-BD	1150.30	2263.09	3454.47	> 3600	> 3600
4	DD-BD	1261.59	2403.79	> 3600	> 3600	> 3600
5	DD-BD	906.34	1863.15	3050.68	> 3600	> 3600

Table 8 Running times (in seconds) of DD-BD for $|V'| = 150$.

Instance #	Model	Number of scenarios				
		50	100	150	200	250
1	DD-BD	2496.16	> 3600	> 3600	> 3600	> 3600
2	DD-BD	2944.20	> 3600	> 3600	> 3600	> 3600
3	DD-BD	2321.62	> 3600	> 3600	> 3600	> 3600
4	DD-BD	2590.34	> 3600	> 3600	> 3600	> 3600
5	DD-BD	2298.36	> 3600	> 3600	> 3600	> 3600