Cross-Dock Trailer Scheduling with Workforce Constraints: A Dynamic Discretization Discovery Approach

Ritesh Ojha, Alan Erera
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia
rojha8@gatech.edu, alan.erera@isye.gatech.edu

LTL freight carriers operate consolidation networks that utilize cross-docking terminals to facilitate the transfer of freight between trailers and enhance trailer utilization. This research addresses the problem of determining an optimal schedule for unloading inbound trailers at specific unloading doors using teams of dock workers. The optimization objective is chosen to ensure that outbound trailers are loaded with minimal delay with respect to their loading deadlines. Formulating this problem, which is known to be NP-hard, using a typical time-expanded network often results in an excessively large mixed-integer programming (MIP) model. To overcome this challenge, we propose an exact dynamic discretization discovery (DDD) algorithm that iteratively solves MIPs formulated over partial networks. The algorithm employs a combination of simple time discretization refinement strategy to progressively refine the partial network until a provably optimal solution is obtained. We demonstrate the effectiveness of the algorithm in solving problem instances representative of a large L-shaped cross-dock in Atlanta. The DDD algorithm outperforms solving the model formulated over a complete time-expanded network with a commercial solver in terms of both computational time and solution quality for practical instances with 180 trailers, 44 unloading doors, and 57 loading doors. Additionally, we compare the DDD algorithm with a state-of-the-art interval scheduling approach using instances from a previous study with a different objective function and additional constraints. The DDD algorithm is computationally faster for most of the small and medium instances and achieves competitive bounds for the larger instances.

Key words: freight transportation, cross-docking, trailer scheduling, dynamic discretization discovery, integer-programming
History:

1. Introduction

The e-commerce industry remains in a rapid growth phase and is forecasted to reach a global market size of US200 billion in total sales by 2026 IIMA (2021). This growth has spurred demand for both parcel and less-than-truckload (LTL) freight services, and carriers providing these services compete by improving shipment delivery speed and reliability. *LTL freight carriers* operate consolidation

networks that use cross-docking terminals to enable freight transfer between trailers; cross-dock consolidation allows carriers to move trailers with higher utilization and to take advantage of trailer cost scale economies Bartholdi III and Gue (2000). A transportation plan or *load plan* specifies the destinations of trailers to be loaded at each cross-dock during each sorting period (or sort), and how shipments are to be routed into loaded trailers (loads) to move between their origins and destinations. Figure 1 illustrates how shipments with different destinations are unloaded from inbound trailers and routed into outbound trailers to some intermediate hubs.

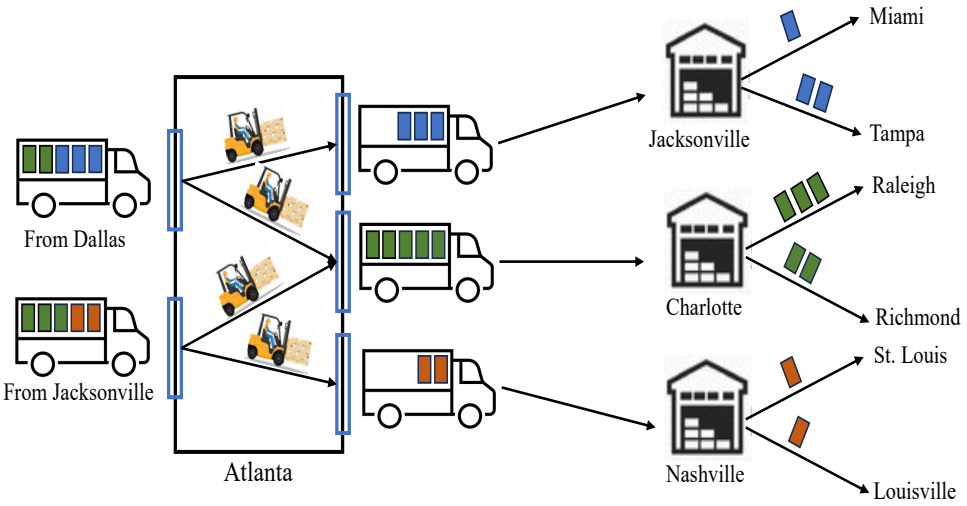


Figure 1 **An illustration of operations at a cross-dock terminal in a middle-mile LTL service network**

Each cross-dock terminal has a set of *unloading doors* (truck docks) and a set of *loading doors*, and shipments in arriving trailers are moved according to the load plan from unloading doors to outbound trailers at loading doors. During a sort, the trailers loaded at an individual loading door will have the same terminal destination. While it is possible to change door destinations from day to day, it is typical in practice to keep terminal-to-door assignments fixed over time and to adjust only when shipment demand and/or the load plan change significantly. Consider now the unloading and loading operation at a cross-dock during a single sort. Inbound trailers for unloading have arrived at the trailer yard by the commencement of the sort or will arrive soon. Outbound (empty) trailers have been staged for loading at each of the loading doors. To initiate the sort, some inbound trailers are moved to specific unloading doors. An individual inbound trailer at an unloading door is worked by logistics workers who unload its shipments and move them to the appropriate trailers at loading doors. Once a trailer is emptied, it is returned to the yard, and the unloading door can be assigned to another inbound trailer. Similarly, once an outbound trailer is filled, it can be moved to the yard for dispatch and replaced at its loading door with an empty trailer. The loading

activities at each loading door are to be completed by a loading deadline which enables trailers for this terminal destination to depart on time. For a more detailed review of cross-dock operations in practice, we refer interested readers to Ladier and Alpan (2016).

A team of logistics workers handles all of the unloading and loading tasks during a sort; typically, these *dock workers* use forklifts or pallet jacks to move shipments between trailers. Once a trailer begins unloading, a dock worker dedicated to this work completes the unloading and transfer of all shipments via a number of round-trips. To speed unloading of some trailers, a larger team of workers can be assigned. Due to the size of the trailers, however, unloading a trailer with more than two workers results in decreasing marginal performance improvements as mentioned in Tadumadze et al. (2019).

Trailer scheduling is the problem of determining a schedule for unloading inbound trailers at specific unloading doors using appropriately-sized teams of dock workers. In this research, we develop an optimization model denoted the cross-dock trailer scheduling problem with workforce constraints (XDTS-W). Given inbound trailers with known planned arrival times for a specific sort at a cross-docking terminal, this decision model determines which trailers to unload at which doors, in which sequence, and with how many dock workers. An objective is chosen to ensure that outbound trailers are loaded with minimal delay with respect to their loading deadlines. As we will show, XDTS-W is a type of nonpreemptive unrelated parallel machine scheduling problem with release dates and an objective function that is the sum of loading task latenesses. Formulating this *NP*-hard problem using a typical time-expanded network often leads to a very large mixed-integer programming (MIP) model that cannot be solved directly using off-the-shelf solvers. Therefore, to solve practical instances motivated by the cross-docking terminals operated by our industry research partner, we propose an iterative exact algorithm that uses a new dynamic discretization discovery approach.

The specific contributions in this paper are as follows.

1. We formulate a MIP model of XDTS-W over a complete time-expanded network. We formulate another MIP model over a partial time-expanded network (XDTS-W-LB) which uses careful modeling of the processing time of unloading trailers to ensure that its objective function value is a lower bound on the optimal objective value of XDTS-W.
2. We develop an exact dynamic discretization discovery (DDD) algorithm for XDTS-W instances and show that the algorithm converges in a finite number of iterations.
3. We demonstrate the effectiveness of the DDD algorithm on instances generated from real-world data from our research partner's cross-dock in Atlanta. The algorithm's performance is influenced by the time window during which all the trailers arrive at the cross-dock. For wider time windows, the DDD algorithm is computationally fast and provides optimal solutions.

When time windows are narrower, the DDD algorithm *often* produces better feasible solutions than directly using commercial solvers.

4. We show that the DDD algorithm also performs well on a slightly different crossdock trailer scheduling problem studied in Tadumadze et al. (2019). For small and medium instances, the DDD algorithm requires far less computational time than the exact interval scheduling approach. For large instances that cannot be solved to optimality by either approach, the DDD algorithm produces the same lower bounds and feasible solutions with similar objective function values.

The rest of the paper is organized as follows. In Section 2, we review relevant literature on cross-dock trailer scheduling, including worker constraints and its relation to parallel machine scheduling. We also provide a brief review on the development of dynamic discretization discovery algorithms. Section 3 presents the formalization and description of XDTS-W, along with its underlying assumptions and a complete time-indexed formulation. The solution methodology for XDTS-W is discussed in Section 4. Section 5 introduces a greedy heuristic for comparative evaluation of the DDD algorithm. Computational studies on instances derived from real-life data and those from Tadumadze et al. (2019) are presented in Section 6 and 7, respectively. Finally, section 8 summarizes the paper's findings and proposes future research directions.

2. Literature Review

Cross-dock Trailer Scheduling. Cross-dock trailer scheduling is closely related to unrelated parallel machine scheduling where trailers are jobs and unloading doors correspond to machines (Tadumadze, Emde, and Diefenbach 2020). The machines (unloading doors) in the cross-docking context are unrelated because each job (trailer) may have a different processing time on each machine because of the relative distance between the unloading door and the loading door destinations of the trailer shipments. We refer interested readers to Boysen and Fliedner (2010) for a detailed classification of the trailer scheduling problems.

In real-world operations, inbound trailers may arrive at various times during the planning horizon, necessitating the consideration of planned arrival times when scheduling trailer unloading at cross-docks (Boysen, Briskorn, and Tschöke 2013). While loading doors are assigned fixed destinations during tactical planning, inbound trailers from different locations can be scheduled at any unloading door (Boysen, Briskorn, and Tschöke 2013, Liao, Egbelu, and Chang 2013). To address the trailer scheduling problem, the authors in Boysen, Briskorn, and Tschöke (2013) develop heuristics that leverage the natural separation between two key decisions: trailer-to-unloading-door assignment and trailer sequence at each unloading door. The authors in Gaudioso, Monaco, and Sammarra (2021) propose a Lagrangian decomposition scheme to minimize the total time required

for unloading and cross-docking shipments to loading doors. In the postal service industry, Boysen, Fedtke, and Weidinger (2017) transform the trailer scheduling problem to an interval scheduling problem and propose decomposition heuristics for the problem. Various other studies (Liao, Egbelu, and Chang 2013, Miao, Lim, and Ma 2009) have focused on comparing the performance of meta-heuristics in tackling the trailer scheduling problem.

Often, the processing time of trailers at unloading doors can be reduced by allocating more workers or resources. Extensive research on this topic in machine scheduling problems is covered in Shabtay and Steiner (2007) and Edis, Oguz, and Ozkarahan (2013). In the cross-docking context, it is challenging and capital-intensive to increase the number of doors to process any arbitrary set of inbound shipments while still meeting service deadlines. Therefore, workers represent the discrete and limited resource used to expedite the unloading process. Several existing studies assume a fixed number of workers for unloading trailers at each door. Shakeri et al. (2012) and Hermel et al. (2016) assign one worker to unload trailers at each door. In real-life operations, terminal managers have the flexibility to determine the number of workers assigned to unload each inbound trailer, and workers can switch from one door to another during the planning period. Many survey papers (Van Belle, Valckenaers, and Cattrysse 2012, Ladier and Alpan 2016, Buijs, Vis, and Carlo 2014) emphasize the significance of workforce scheduling to unload trailers at a cross-dock. The authors in Tadumadze et al. (2019) solve the integrated trailer and workforce scheduling problem with a fixed number of workers available during the planning horizon. Corsten, Becker, and Salewski (2020) propose an optimization model to integrate trailer and workforce scheduling with permanent and temporary workers to facilitate the unloading and loading activities.

To the best of our knowledge, the current state-of-the-art methodology to solve the integrated trailer and workforce scheduling problem is presented in Tadumadze et al. (2019). In their work, the authors introduce an interval scheduling formulation with a predefined integer parameter $\mu \geq 1$ to regulate the number of potential intervals for unloading inbound trailers. Solving the interval scheduling formulation using a commercial solver yields an exact approach when $\mu = 1$ since the formulation includes all possible time intervals for processing inbound trailers. However, when $\mu > 1$, some intervals are dropped which reduces the solution space and yields a heuristic solution.

This study differs from Tadumadze et al. (2019) in terms of the problem and solution approach. First, we do not have strict deadlines for unloading inbound trailers; we will demonstrate in Section 7 that our approach is effective even when there are hard deadlines. Second, our decision to unload a trailer at a specific door is dependent on both the door location and the number of assigned workers. Finally, we develop an iterative exact DDD algorithm that progressively adds new time points to a time-expanded network model (to improve the lower-bound) until a provably optimal solution is obtained. This approach may be more flexible than solving the model with all possible start time points/processing intervals ($\mu = 1$) as in Tadumadze et al. (2019).

Dynamic Discretization Discovery. Boland et al. (2017) showed that provable continuous-time optimal solutions for service network design problems can be obtained by iteratively refining MIP models formulated over a partial time-expanded network. Marshall et al. (2020) developed an interval-based DDD algorithm which was shown to be more effective in solving larger instances efficiently. Scherr et al. (2020) consider an extended service network design problem that includes mixed autonomous fleets and proposed an improved DDD algorithm that uses valid inequalities to strengthen the lower bounds. Hewitt (2019) proposed an enhanced DDD algorithm to solve the continuous-time load plan design problem in LTL freight transportation. The authors propose algorithmic enhancements to reduce the number of integer programs solved by the algorithm; these enhancements include solving a relaxed model to generate the partial network, adding valid inequalities in each iteration, and introducing symmetry-breaking procedures to determine how a partial network is refined in each iteration. Recently, He et al. (2022a) developed an exact DDD algorithm for the service network design problem with hub capacities and He et al. (2022b) for the minimum duration time-dependent shortest path problem with piece-wise linear travel times. DDD algorithms have also been successfully applied to applications such as the time-dependent traveling salesman problem with time windows (Vu et al. 2020), the time-dependent minimum tour duration problem (Vu, Hewitt, and Vu 2022), and the continuous-time inventory routing problem (Lagos, Boland, and Savelsbergh 2020).

3. Cross-dock Trailer Scheduling with Workforce Constraints

3.1. Problem Description

Each cross-dock has a set of unloading doors U and a set of loading doors L. Each loading door ℓ has a specific terminal destination, and the outbound trailers from the loading door have a known departure deadline d_ℓ. A set of inbound trailers I arrive at the cross-dock where each trailer $i \in I$ has a planned arrival time r_i. The planned arrival time of a trailer is analogous to the release times of jobs in machine scheduling. Upon arrival, these inbound trailers can be directed to an available door for unloading. Unloaded shipments are cross-docked using forklifts (see Figure 1) directly to their corresponding loading doors. Each forklift carries a single shipment at a time and is operated by one worker. A trailer completes unloading when all of its shipments have been cross-docked to their respective loading doors.

The total processing time of an inbound trailer in a cross-dock includes (i) docking at door (ii) preparation for unloading (iii) unloading and (iv) undocking from door. The unloading time of a trailer can be reduced by allocating multiple workers to unload the trailer. However, it is important to note that allocating more workers to unload a trailer does not result in a proportionate improvement in performance due to space limitations within the trailer (Tadumadze et al. 2019).

We denote the set of permissible worker allocations for unloading a trailer as M, and Q represents the total number of workers available during the planning period. Once assigned to unload a trailer, workers remain busy from the start until the completion of the unloading process. Instead of directing workers to unload parts of multiple trailers, it is more practical and straightforward to reassign them to another trailer once a trailer has been completely unloaded (Tadumadze et al. 2019). We assume that workers can transition from one unloading door (after finishing unloading the current trailer) to another unloading door instantaneously. This assumption is reasonable since the time required for walking or driving (in the case of a forklift) between doors is significantly shorter than the unloading time.

Terminal managers have access to information such as the shipments in a trailer, shipment destinations, and planned arrival times of trailers. Let S_i be the set of shipments on trailer $i \in I$, D_i be the set of loading doors associated with shipments on trailer $i \in I$. Let $n_{i\ell}$ be the number of shipments on trailer i that need to be cross-docked to loading door ℓ, and $\tau_{u\ell}$ be the total time for a worker to transport a shipment from unloading door u to loading door ℓ and then return to the unloading door.

We use a *conservative* modeling approach to estimate the total processing time of trailers. In this approach, an inbound trailer is completely processed or unloaded when all of its shipments have been transferred to their respective loading doors. When a single dock worker is unloading, the processing time of a trailer at an unloading door is dependent on the total out-and-back travel time, summed over all shipments, between the unloading door and loading doors. These travel times may be affected by terminal congestion, but for simplicity we ignore such effects and assume instead that travel times do not depend on terminal load or scheduling decisions. The processing time of trailer i at door u using one worker is denoted by ρ_u^{i1} and can be calculated as shown in (1a). We define the processing time of a trailer with m workers as shown in (1b), where $\kappa \geq 1$ is a constant and represents the acceleration effect of additional workers (Tadumadze et al. 2019) and $\lfloor . \rceil$ function rounds the value to the nearest integer. Increasing κ to an integer value larger than one results in *sub-additive improvement* in the unloading time.

$$\rho_u^{i1} = \sum_{\ell \in D_i} n_{i\ell}\tau_{u\ell} + \text{docking and undocking time} \quad \forall i \in I, u \in U \tag{1a}$$

$$\rho_u^{im} = \left\lfloor \frac{\rho_u^{i1}}{m^{1/\kappa}} \right\rceil \quad \forall i \in I, u \in U, m \in M \tag{1b}$$

If a trailer i starts unloading at time t, then it completes unloading at time $t + \rho_u^{im}$ and the m workers assigned to the trailer are busy during the interval $[t, t + \rho_u^{im})$.

Given these inputs, optimization problem XDTS-W seeks to schedule specific inbound trailers for unloading at specific unloading doors over time with an assigned number of workers. The objective is to minimize the total deadline violation of the outbound trailers at the loading doors. The presence of shipments bound for multiple outbound trailers within each inbound trailer adds significant complexity to the problem, since the decision to schedule an inbound trailer at some unloading door at some start time with some workers then impacts the earliest departure time for many loading trailers. The planning horizon for such a problem might vary from one sorting period (for example, the sunrise sort from *3am-6am*) to multiple sorts (for example, twilight and night sorts from *5pm-3am*).

3.2. A Mixed-Integer Programming Formulation

XDTS-W can be modeled using either compact or extended mixed-integer programming scheduling formulations. Compact formulations require "big-M" constraints that lead to weak linear-programming relaxations of the MIP, and thus such formulations cannot be used to solve large instances in reasonable times (Lagos, Boland, and Savelsbergh 2022). Extended formulations, on the other hand, have tighter linear-programming relaxations. This section presents an extended formulation MIP model for XDTS-W using a time-expanded network denoted by $\widehat{G} = (\widehat{N}, \widehat{A})$ where $\widehat{N}$ denotes the set of nodes and $\widehat{A}$ denotes the set of arcs. Each node $(u,t) \in \widehat{N}$ represents an unloading door $u \in U$ at time t. Nodes are defined at each unloading door u for all possible time points during planning horizon T; therefore each door has the same set of time points. To define all possible time points, suppose that T is discretized at the Δ-minute-level, where $\Delta = GCD(GCD_{i\in I} r_i, GCD_{\ell\in L} d_\ell, GCD_{(i,m,u)} \rho_u^{im})$ and GCD(.) is the greatest common divisor function; let $\widehat{\mathcal{T}}$ be the set of unique time points in the planning horizon. Furthermore, assume that all problem parameters that measure time $(r_i, d_\ell, \rho_u^{im})$ are measured in units of Δ. Given a complete set of time-space nodes, suppose that $\widehat{A}$ includes an arc generated from $(u,t) \in \widehat{N}$ to $(u, t+\rho_u^{im}) \in \widehat{N}$ for all trailers $i \in I$, for all possible values of m, and for all nodes $(u,t) \in \widehat{N}$, such that $r_i \le t$. With such a definition and the fact that the processing time is ρ_u^{im}, an arc $((u,t),(u,t+\rho_u^{im}))$ can be used to represent the decision to begin unloading some trailer i with m workers at door u at time t, thus occupying the door until time $t+\rho_u^{im}$ when the trailer unloading is complete.

Let binary decision variable x_{ut}^{im} take value 1 if trailer i starts unloading with m workers at door u at time t, where $r_i \le t$, and take value 0 otherwise. This decision can be interpreted as an assignment a_{ut}^{im} of trailer i with m workers to time-space occupation arc $((u,t),(u,t+\rho_u^{im}))$. Continuous decision variable z_ℓ measures the deadline violation at loading door ℓ. Consider then the MIP formulation shown in Model 2:

$$\text{Minimize} \quad \sum_{\ell\in L} z_\ell \tag{2a}$$

$$\text{s.t.} \quad \sum_{m\in M}\sum_{\substack{(u,t)\in\widehat{N}\\ t\geq r_i}} x_{ut}^{im} = 1, \quad \forall\ i\in I \tag{2b}$$

$$\sum_{m\in M}\sum_{\substack{(u,t)\in\widehat{N}\\ t\geq r_i}} \left(t+\rho_u^{im}\right) x_{ut}^{im} - z_\ell \leq d_\ell, \quad \forall\ i\in I, \ell\in D_i \tag{2c}$$

$$\sum_{a_{ut'}^{im}\in\widehat{\mathcal{A}}_{ut}} x_{ut'}^{im} \leq 1, \quad \forall\ (u,t)\in\widehat{N} \tag{2d}$$

$$\sum_{u\in U}\sum_{a_{ut'}^{im}\in\widehat{\mathcal{A}}_{ut}} m x_{ut'}^{im} \leq Q, \quad \forall\ t\in\widehat{\mathcal{T}} \tag{2e}$$

$$x_{ut}^{im}\in\{0,1\}, \quad \forall\ i\in I, m\in M, (u,t)\in\widehat{N}, t\geq r_i \tag{2f}$$

$$z_\ell \geq 0, \quad \forall\ \ell\in L \tag{2g}$$

We refer to this model as XDTS-W. In this model, let $\widehat{\mathcal{A}}_{ut}$ represent all possible (arc) assignments that occupy unloading door u at time t,

$$\widehat{\mathcal{A}}_{ut} \equiv \{a_{ut'}^{im} : t-\rho_u^{im} < t' \leq t\,,\, ((u,t'),(u,t'+\rho_u^{im}))\in\widehat{A},\, i\in I,\, m\in M\} \quad . \tag{3}$$

The objective function in (2a) minimizes the total penalty due to deadline violation at loading doors. Constraints (2b) ensure that a trailer starts processing after its planned arrival time and at an unloading door at exactly one time point using a certain number of workers. Constraints (2c) determine the loading door deadline violations by ensuring that the maximum of the actual completion time or the deadline for loading door ℓ is not less than the completion time implied by any unloading trailer i sending shipments to ℓ. Constraints (2d) ensure that at most one trailer can be processed at an unloading door at a given time, similarly to those found in time-indexed formulations for single machine scheduling problems in Sousa and Wolsey (1992). Constraints (2e) ensure that the maximum worker availability is respected at all times. Constraints (2f) and (2g) define the domain and range of variables. Note that if the loading door deadlines are all large enough, then every feasible solution is optimal.

Theorem 1 *XDTS-W is strongly NP-hard.*

Proof See Appendix 9.1

4. Solution Methodology

Model 2 has a significant drawback for many large, realistic instances. Since such instances have large numbers of time points for many unloading doors, the number of decision variables and constraints can also grow very large, which may lead to intractable mixed-integer programs. Note that the total number of decision variables in XDTS-W is $O(|I||U||M||\widehat{\mathcal{T}}|)$ and the total number

of constraints is $O(|I|+|I||L|+|U||\widehat{\mathcal{T}}|+|\widehat{\mathcal{T}}|)$. This paper proposes a solution methodology with a scheme to alleviate this difficulty by using *dynamic discretization discovery* ideas. In this scheme, we begin with a smaller formulation that uses a partial network that contains only a small subset of the nodes in the complete time-expanded network. Importantly, we show that solving the formulation using the partial network produces a lower bound on the objective function value to XDTS-W; we will refer to this model therefore as XDTS-W-LB. Furthermore, we show that it is simple to determine whether a feasible (or optimal) solution to XDTS-W-LB is also feasible for XDTS-W; such solutions provide upper bounds and can be used to determine the current optimality gap. If an optimal solution has not yet been identified, we show that we can create a new partial network by adding new time points, network nodes, and assignment arcs using a systematic procedure that ensures that the lower-bounding property is preserved while ensuring that the previous solution to XDTS-W-LB is no longer feasible; these iterations can be repeated until an optimal solution is found. We also develop two simple algorithms for converting any feasible solution to XDTS-W-LB to a feasible solution of XDTS-W; these algorithms can be used, periodically, to improve the upper bound, and/or after a time limit if no previous feasible solution has been identified.

Section 4.1 defines the partial network and XDTW-W-LB. Section 4.2 presents approaches to construct feasible solutions to XDTS-W given feasible solutions to XDTS-W-LB. Finally, Section 4.3 summarizes the complete DDD algorithm.

4.1. Relaxation of XDTS-W

Consider a partial network $G=(N,A)$ which consists of nodes $(u,t)\in N\subseteq\widehat{N}$ and arcs A. Let $\mathcal{T}(u)$ denote the set of time points at unloading door $u\in U$ in G. The network G is constructed to satisfy certain properties. These properties ensure that any feasible solution of XDTS-W (with integer arrival times, processing times, and deadlines) can be mapped to a feasible solution in G, and this solution has a cost not greater than the cost of the feasible solution of XDTS-W.

Property 1 *Node set N minimally contains all nodes of type $(u,r_i)\ \forall\ u\in U, i\in I$ where r_i is the planned arrival time of trailer $i\in I$.*

Property 2 *Given node set N, the arc set A contains an arc $((u,t),(u,t+\widehat{\rho}^{im}_{ut}))$ from each $(u,t)\in N$ for all unique values of $\widehat{\rho}^{im}_{ut}$ defined as $\max\{\widehat{\rho}:0\leq\widehat{\rho}\leq\rho^{im}_{u}\,,(u,t+\widehat{\rho})\in N\}$ for each trailer $i\in I$ where $t\geq r_i$ and number of workers $m\in M$.*

Property 3 *Each unloading door has the same set of time points, i.e., if $(u,t)\in N$ then $(u',t)\in N$ for all $u'\in U$.*

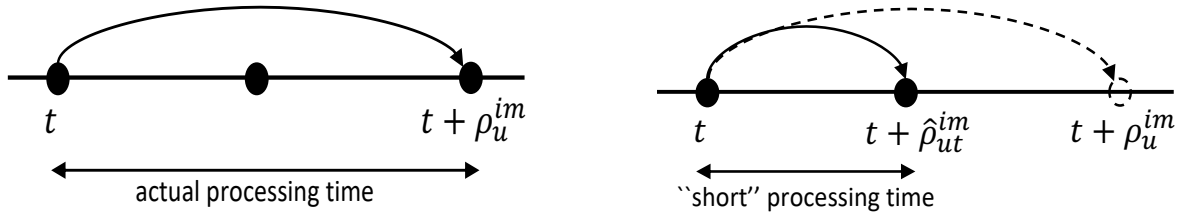
Henceforth, we will use $\mathcal{T}$ to denote the same set of time points at each unloading door in the partial network G as indicated by Property 3.

As in Section 3.2, suppose that we use the notation a_{ut}^{im} to represent the decision to assign trailer i for unloading with m workers to unloading door u at time t. In a partial network, this assignment occupies arc $((u,t),(u,t+\widehat{\rho}_{ut}^{im}))$. Similar to earlier, let $\mathcal{A}_{ut}$ represent all possible (arc) assignments that occupy unloading door u at time t but now in the partial network G, that is $\mathcal{A}_{ut} \equiv \{a_{ut'}^{im} : t - \widehat{\rho}_{ut'}^{im} < t' \leq t\ ,\ ((u,t'),(u,t'+\widehat{\rho}_{ut'}^{im})) \in A,\ i \in I,\ m \in M\}$ in the partial network. Note that $\mathcal{A}_{ut}$ is obtained for the partial network G by replacing ρ with $\widehat{\rho}$ and $\widehat{A}$ with A in (3). To obtain a relaxation of XDTS-W, we now simply modify Model (2) by replacing $\widehat{N}$ with N, $\widehat{A}$ with A, $\widehat{\mathcal{T}}$ with $\mathcal{T}$ and $\widehat{\mathcal{A}}$ with $\mathcal{A}$. Importantly, note that the objective function value as defined by (2a) and (2c) continues to use the original parameter ρ_u^{im}. We refer to the resulting model as XDTS-W-LB.

Given this definition of XDTS-W-LB from partial network G, Properties 1 and 2 ensure that a possible assignment exists that begins unloading trailer i at door u at its arrival time with some workers. Furthermore, Property 2 ensures that a trailer assignment to an arc in G creates a so-called *optimistic mapping* of the associated unloading completion time of that trailer. Note that the actual completion time for trailer i given assignment a_{ut}^{im} is $t+\rho_u^{im}$, but Property 2 ensures that i is modeled to occupy door u only until time $t+\widehat{\rho}_{ut}^{im}$. Recall that $\widehat{\rho}_{ut}^{im} \leq \rho_u^{im}$ and the inequality is strict when $t+\rho_u^{im} \notin \mathcal{T}$. Thus, trailer assignments using the arcs of G with the same number of workers lead to trailer completion times that are not later than the actual completion times which result from using the arcs of the full network $\widehat{G}$. It follows that the unloading door u following the assignment a_{ut}^{im} when using G is modeled to be available not later than it would be when using $\widehat{G}$, and this property is the key to ensuring that XDTS-W-LB produces lower bounds to the optimal solution of XDTS-W.

Figure (2) provides a graphical depiction of the optimistic mapping of the completion time of trailer i to $t+\widehat{\rho}_{ut}^{im}$ if the actual completion time $t+\rho_u^{im} \notin \mathcal{T}$. In this case, we say that trailer i uses a "short" processing time arc when unloading at (u,t). Note that a trailer can be modeled in XDTS-W-LB with $\widehat{\rho}_{ut}^{im} = 0$ if there does not exist any time point $t' \in \mathcal{T}$ such that $t < t' \leq t+\rho_u^{im}$; in this case the trailer starts and completes at the same time t.

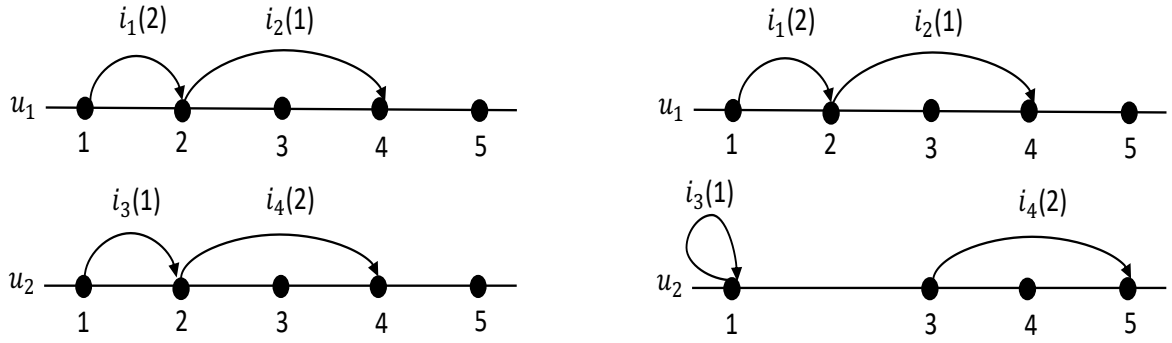
Finally, Property 3 ensures that each unloading door has the same set of time points in the partial network; when G does not satisfy this property, we show that XDTS-W-LB may not yield a lower bound to XDTS-W. Consider the example shown in Figure 3, with 3 workers, two unloading doors u_1, u_2 and four trailers i_1, i_2, i_3, i_4. Suppose there is a single loading door with deadline of 4 time units. Let the processing time of trailers i_1, i_2 at door u_2 and the processing time of trailers i_3, i_4 at door u_1 be a very large number; therefore, trailers i_1 and i_2 are scheduled at door u_1 and trailers i_3 and i_4 are scheduled at door u_2 in an optimal solution. Now suppose, trailers i_1 and i_2 are



(a) Actual processing time arc in $\widehat{G}$ (b) "Short" processing time arc in G

Figure 2 Processing time arcs in complete ($\widehat{G}$) and partially (G) time expanded network

unloaded using two and one worker, respectively, at door u_1. The values in the brackets denote the number of workers used to unload the trailer. Similarly, trailers i_3 and i_4 are unloaded using one and two workers, respectively, at door u_2. Figure 3a depicts an optimal solution in the complete time-expanded network; the objective function value is 0 since all of the trailers complete unloading by time 4. Figure 3b represents an optimal solution over a partial network where time point $t = 2$ is missing from door u_2. By the definition of $\widehat{\rho}$, trailer i_3 starts unloading at time $t = 1$ and door u_2 becomes instantaneously available for another trailer unloading. However, trailer i_4 cannot start unloading at time $t = 1$ with 2 workers because only 1 is available until $t = 2$. Importantly, trailer i_4 cannot be unloaded at $t = 2$ because the time point does not exist at u_2. Thus, trailer i_4 starts unloading at $t = 3$ resulting in a deadline violation and an optimal objective function value of 1.



(a) trailer schedule in complete network $\widehat{G}$ (b) trailer schedule in partial network G

Figure 3 Example illustrating why Property 3 is necessary for G to ensure that XDTS-W-LB yields a lower-bound to XDTS-W

To show that the optimal solution of XDTS-W-LB yields a lower-bound to the optimal solution of XDTS-W, it is sufficient to show that any feasible solution of XDTS-W can be mapped to a feasible solution of XDTS-W-LB with an objective function value that is not larger than its objective value for XDTS-W. Given a feasible solution $(\widehat{x}, \widehat{z})$ to XDTS-W, we define a mapping to

construct x-variables in XDTS-W-LB as shown in (4). Note that for any given time point $t \in \mathcal{T}$, $t^+ = \min\{t' : t' > t, t' \in \mathcal{T}\}$ is the next time point in the discretization.

$$x_{ut}^{im} = \sum_{\substack{t' \in \widehat{\mathcal{T}} \\ t' \in [t, t^+)}} \widehat{x}_{ut'}^{im} \quad \forall\, i \in I, m \in M, (u,t) \in N \tag{4}$$

Definition (4) sets a binary variable in XDTS-W-LB as the sum of binary variables corresponding to arc assignments that start unloading trailer $i \in I$ with $m \in M$ workers at door $u \in U$ in the interval $[t, t^+)$ in XDTS-W, thus mapping these start times from the full network to the time $t \in \mathcal{T}$ in the partial network. We will prove that the resulting solution x is always feasible for XDTS-W-LB when G satisfies Properties 1-3.

Lemma 1 *Given a feasible solution to XDTS-W with non-zero processing times of trailers, if the start times of $q > 1$ trailers unloaded at any door u lie in the interval $[t, t^+)$ and the start times are mapped to the common time t by (4), then at least $q-1$ of the trailers also complete unloading at time t in the partial network G.*

Proof See Appendix 9.2

Lemma 1 implies that multiple trailers can start unloading at a door at a given time point in a partial network but at most one of them can have non-zero processing time; see Appendix 9.3 for an example.

Theorem 2 *XDTS-W-LB defined using partial network G satisfying Properties 1-3 is a relaxation of XDTS-W and yields a lower bound to the objective function value of XDTS-W.*

Proof See Appendix 9.4 □

4.2. Constructing Feasible Solutions to XDTS-W from a Feasible Solution to XDTS-W-LB

A solution to XDTS-W-LB can be used directly to identify a feasible solution to XDTS-W. Consider a feasible set of assignments $\{a_{ut}^{im}\}$ after solving XDTS-W-LB. Each assignment $\{a_{ut}^{im}\}$ where $x_{ut}^{im} = 1$ represents starting the unloading of trailer i with m workers at door u at time t and occupying that door for $\widehat{\rho}_{ut}^{im}$ time (using the occupation arc $((u,t),(u,t+\widehat{\rho}_{ut}^{im})) \in A$). If $\widehat{\rho}_{ut}^{im} < \rho_u^{im}$, then we say that trailer i has been assigned to a short processing time arc. If no trailer i is assigned to a short processing time arc, then the feasible solution to XDTS-W-LB is feasible for XDTS-W. Furthermore, if this feasible solution is also optimal for XDTS-W-LB, then it is also optimal for XDTS-W.

In most iterations, however, solving XDTS-W-LB does not lead to such a direct feasible solution to XDTS-W. Therefore, it may be useful to have alternative approaches to create feasible solutions

to potentially improve an upper bound on the optimal objective to XDTS-W. In this section, we define two procedures to construct such solutions directly from feasible (or optimal) solutions to XDTS-W-LB. Both approaches will fix the trailer-to-door assignments, number of workers per trailer, and trailer sequence decisions for each unloading door from the XDTS-W-LB solution and then create a feasible solution that respects the total workforce constraint.

The first approach uses a greedy idea. Consider a set of optimal assignment decisions $\{x^{*im}_{ut}\}$ after solving XDTS-W-LB. For each trailer i, there is a unique triplet (u, m, t) where $x^{*im}_{ut} = 1$; trailer i starts unloading at door u with m workers at time t in the lower bound solution. A feasible solution can be generated by fixing the u and m decisions for each trailer and then applying a simple delaying scheme to select a feasible unloading start time $\widehat{t}$. To do so, consider the trailers in non-decreasing order of start times t. Then, for each trailer in this ordering, set $\widehat{t} \geq t$ to be the earliest time that door u is available and m workers are available. When trailer i is started at $\widehat{t}$, then door u is marked occupied until $\widehat{t} + \rho^{im}_u$. Furthermore, the available worker pool is reduced by m at time $\widehat{t}$ and then increased by m at $\widehat{t} + \rho^{im}_u$. It should be clear that such an approach ensures that constraints (2d) and (2e) for XDTS-W are satisfied by the constructed solution.

The second approach recognizes that the greedy approach for delaying trailer start times can be improved if the allocation of dock workers to unloading activities is jointly optimized; appendix 9.5 presents an example to illustrate this observation. Consider then a simple mixed-integer programming model to do so that we refer to as the worker dispatch model (WDM). In this worker flow model, we imagine that Q workers are initially available (at a source node) and can be dispatched to unload trailers at specific start times. Similar to above, if m workers are assigned to trailer i at unloading door u at start time t, then those m workers become available for reassignment at time $t + \rho^{im}_u$.

To complete the formulation of WDM, consider an optimal solution to XDTS-W-LB where (u_i, m_i, t_i) is the triplet indicating that trailer i is unloaded at door u_i with m_i workers at the proposed time t_i. To preserve the unloading order at each door, let parameter $\gamma_{ik} = 1$ if trailer i is unloaded before trailer k at the same door ($t_i < t_k$ and $u_i = u_k$) and 0 otherwise for all pairs of trailers $i, k \in I$. Also, for ease of notation define a trailer processing time parameter $p_i = \rho^{im_i}_{u_i}$ also for each $i \in I$. Integer decision variables $v_{ik} \in \mathbb{Z}_{\geq 0}$ are used in the formulation to determine the number of workers who next unload trailer k after currently unloading trailer i; note that trailer $i = 0$ is used to indicate a dummy source such that v_{0k} counts the number of workers whose first assignment is to trailer k. Binary decision variables y_{ik} must be set to one if at least one worker unloads trailer k after unloading trailer i. Continuous variables $z_\ell \in \mathbb{R}_{\geq 0}$ determine the deadline

violation at loading door $\ell \in L$, while continuous variables $s_i \in \mathbb{R}_{\geq 0}$ denote the unloading start time of trailer $i \in I$. Consider now the following formulation:

$$\text{Minimize} \sum_{\ell \in L} z_\ell \tag{5a}$$

$$s_i \geq r_i \quad \forall\ i \in I \tag{5b}$$

$$s_i + p_i - z_\ell \leq d_\ell \quad \forall\ i \in I, \ell \in D_i \tag{5c}$$

$$s_i + p_i - \mathcal{M}\,(1 - y_{ik}) \leq s_k \quad \forall\ i \in I, k \in \{k' \in I : \gamma_{ik'} \neq 1\}, \tag{5d}$$

$$s_i + p_i \leq s_k \quad \forall\ i \in I, k \in \{k' \in I : \gamma_{ik'} = 1\} \tag{5e}$$

$$\sum_{i \in I} v_{0i} \leq Q \tag{5f}$$

$$v_{ki} \leq Q y_{ki} \quad \forall\ i, k \in I \tag{5g}$$

$$\sum_{k \in I \cup \{0\}} v_{ki} \geq \sum_{k \in I} v_{ik} \quad \forall\ i \in I \tag{5h}$$

$$\sum_{k \in I \cup \{0\}} v_{ki} \geq m_i \quad \forall\ i \in I \tag{5i}$$

$$v_{ik} \in \mathbb{Z}_{\geq 0} \quad \forall\ i, k \in I \tag{5j}$$

$$v_{0i} \in \mathbb{Z}_{\geq 0} \quad \forall\ i \in I \tag{5k}$$

$$y_{ik} \in \{0, 1\} \quad \forall\ i, k \in I \tag{5l}$$

$$s_i \geq 0 \quad \forall\ i \in I \tag{5m}$$

$$z_\ell \geq 0 \quad \forall\ \ell \in L \tag{5n}$$

The objective function in (5a) minimizes the total deadline violation at the loading doors. Constraints (5b) ensure that the trailers start unloading after they have arrived at the cross-dock. Constraints (5c) determine the loading door deadline violation. If at least one worker unloads trailer k after unloading trailer i and $u_i \neq u_k$, then constraints (5d) ensure that trailer k starts unloading after trailer i has completed unloading. If trailer k is sequenced after trailer i at the same unloading door, then constraints (5e) ensure that trailer k is unloaded after trailer i. Constraints (5f)-(5h) conserve the flow of workers and ensure that at most Q workers are used. Note that constraints (5h) have inequality instead of equality to allow workers to stay at the door where the current trailer is being unloaded, if there are no more trailers to unload after the current trailer. Constraints (5i) ensure that the trailers are unloaded using m_i workers. Constraints (5j)-(5n) define the domain and range of variables. Note that the formulation permits a flow of more than m_i workers from trailer k to trailer i. However, these additional workers do not impact the processing time p_i. Note also that the formulation also prevents workers moving backwards in time from a later trailer to an earlier trailer at the same loading door.

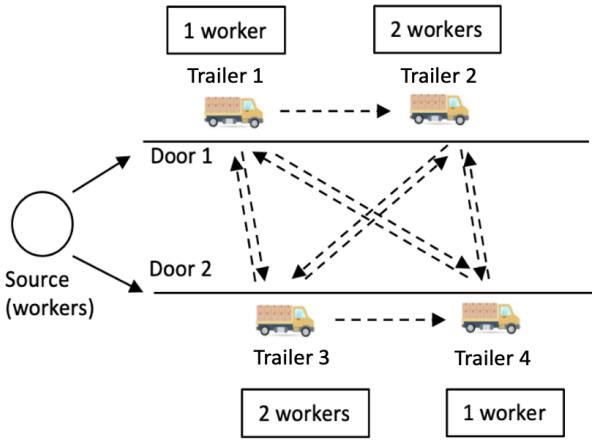


Figure 4 **Example illustrating dispatch of workers to unload trailers**

Figure 4 presents an example to illustrate the dispatch of three workers from the source node to unload trailers. Trailer 1 is unloaded by one worker and then trailer 2 is unloaded by two workers at unloading door 1. Similarly, trailer 3 is unloaded by two workers and then trailer 4 is unloaded by one worker at unloading door 2. A feasible solution to WDM represents the flow of three workers as $source \rightarrow trailer1 \rightarrow trailer2$, $source \rightarrow trailer3 \rightarrow trailer4$, $source \rightarrow trailer3 \rightarrow trailer2$. Note in this case that the unloading time of Trailer 2 cannot be earlier than the completion time of Trailer 3 given this worker flow.

Importantly, the construction of WDM leads to a clear observation. Suppose that an optimal solution to XDTS-W is such that the triplet (u_i, m_i, t_i) is associated with each trailer $i \in I$. For this triplet, u_i is the optimal unloading door for trailer i and m_i is its optimal number of workers. However, suppose that the values t_i only specify an optimal *ordering* of trailers at their unloading doors (when trailers are unloaded in order of non-decreasing t_i) but not the actual optimal unloading start times. Then, WDM will determine the actual optimal unloading times s_i and its optimal objective function value will equal z^*, the optimal objective function value for XDTS-W.

4.3. Dynamic Discretization Discovery (DDD) Algorithm

We now summarize a complete DDD algorithm for solving XDTS-W either to optimality or to a provable optimality gap. Again, the core idea of this algorithm will be to repeatedly solve instances of XDTS-W-LB over partial networks that satisfy Properties 1-3, where each new partial network adds new timed nodes (and subsequently builds a new set of timed arcs). A specific systematic procedure adds new timed nodes in each iteration such that at least one assignment of an unloading trailer to an infeasible (short) processing time arc is eliminated from the solution. This procedure ensures that solving XDTS-W-LB produces a sequence of non-decreasing objective function lower bounds. The best lower bound can be compared to the objective function of any feasible solution to

determine a current optimality gap; the approach can be terminated when a target gap is achieved, when a time limit or iteration count is exceeded, or when XDTS-W-LB returns a feasible solution to XDTS-W which is provably optimal.

More specifically, we use Algorithm 1 to find solutions in the computational studies reported in this paper. Appendix 9.6 illustrates the individual steps of the DDD algorithm on a small example instance. Algorithm 1 as specified terminates when either an optimal solution is identified, an iteration count is reached, or a target computational run time is reached; additional or alternative stopping criteria such as reaching a target optimality gap could also be used. In the steps of the algorithm below, note that an optimal solution is found either when XDTS-W-LB is solved optimally during an iteration and none of the trailers $i \in I$ are assigned to short processing time arcs, or when the current best feasible solution identified has an objective function value equal to the best lower bound found. If an optimal solution has not been found, we add additional time points and the associated time-space nodes at all unloading doors to G and regenerate the arc set for the next iteration. We note that this algorithm will terminate with an optimal solution if we always solve XDTS-W-LB to optimality in each iteration. It is possible to implement this approach slightly differently if we cannot solve XDTS-W-LB to optimality.

Theorem 3 *DDD algorithm finds the optimal solution to XDTS-W in at most* $(T+1-\Phi)$ *iterations where* Φ *denotes the number of unique time points in the set* $\{r_i : i \in I\}$.

Proof See Appendix 9.7

5. Greedy Heuristic

In this section, we describe a two-phase greedy heuristic to find a feasible solution to XDTS-W. The first phase schedules trailers at unloading doors, and the second phase assigns workers to unload the trailers. The proposed greedy heuristic will be used as a baseline to compare the effectiveness of solutions generated via optimization.

Trailer Scheduling: In the first stage, all inbound trailers $i \in I$ are sorted in non-decreasing order of their planned arrival times r_i. In order, each trailer i is then assigned to the best loading door u and specific (earliest) unloading time t, where best is defined to minimize the total deadline violation at the loading doors for this trailer's shipments assuming that the trailer is unloaded with one worker. The unloading time t assigned to this trailer is the later of its arrival time r_i and the earliest time door u is available for its next trailer. For example, suppose trailer i has completion time $\bar{C}(u)$ at unloading door $u \in U$ and shipments are cross-docked to one or more loading doors ℓ with deadline d_ℓ. The trailer is assigned to a door u^* as defined in (6):

$$u^* = \arg\min_{u \in U} \sum_{\ell \in D_i} \max\{\bar{C}(u) - d_\ell, 0\} \tag{6}$$

Algorithm 1 DDD Algorithm for XDTS-W Instances

$maxIter \leftarrow$ maximum allowed iterations
$timeLimit \leftarrow$ maximum allowed time for running the DDD algorithm
$G = (N, A) \leftarrow$ a partial network satisfying Properties 1-3
While ($iteration \leq maxIter$ and $runtime \leq timeLimit$)
  Solve XDTS-W-LB using partial network G
  $S \leftarrow$ optimal solution found solving XDTS-W-LB
  $LB \leftarrow$ optimal objective value of S
  If (all trailers assigned to correct processing times in S ($x_{ut}^{im} = 1 \rightarrow \widehat{\rho}_{ut}^{im} = \rho_u^{im}$)): S is an optimal solution to XDTS-W
    break
  Solve WDM using solution S to XDTS-W-LB
  $S_F \leftarrow$ best feasible solution found using WDM during this or any earlier iteration
  $UB \leftarrow$ best objective value of S_F
  If $(LB = UB)$: S_F is an optimal solution to XDTS-W
    break
  Else
    $\forall\ i \in I$ where solution S has $x_{ut}^{im} = 1$ and $\widehat{\rho}_{ut}^{im} < \rho_u^{im}$, add time point $t + \rho_u^{im}$ at every $u \in U$ and associated time-space nodes to N
    Regenerate arc set A given updated node set N
    $iteration \leftarrow iteration{+}1$
    $runtime \leftarrow runtime$ + time expended this iteration
  End If
End While

Worker Assignment: In the second stage, trailers are re-sorted now in non-decreasing order of their scheduled unloading start times determined in the first stage. Let *idle* be the number of workers available when scheduling trailer i at planned time t. If $idle > 0$, then trailer i is assigned the minimum of *idle* workers or the maximum number of workers that can be assigned to any trailer. If $idle = 0$, then the unloading start time t is delayed until the first later time $t' > t$ where $idle > 0$ before receiving an assignment of workers. This greedy approach ensures that workers are never idle at a time when trailers are waiting to be assigned workers.

6. Computational Study

In this section, we compare the performance of the DDD algorithm on practical instances of XDTS-W derived from data provided by a large US LTL carrier research partner. Section 6.1 describes

the instances and parameters. Section 6.2 discusses how the algorithm is tuned to decide when to use WDM to create feasible solutions. Section 6.3 compares the performance of the DDD algorithm to using a commercial solver to solve XDTS-W using the complete time-expanded network and to the greedy heuristic. Section 6.4 examines how the choice of the initial partially-expanded network impacts the performance of the DDD algorithm.

For this study, all algorithms and heuristics were implemented in Python 3.8 and the MIP models were solved using *Gurobi* 9.0. Experiments were run on a local macOS machine with a 6-core 2.6 GHz processor and 16 GB of RAM.

6.1. Description of Instances and Parameters

In this study, we use data from an *L*-shaped cross-dock in Atlanta with 44 unloading doors and 57 loading doors. The actual travel distances between the doors and the average speed of forklifts (driven by workers) are used to calculate the travel time between doors. The planning horizon for the scheduling of trailers is determined by the length of the sorting period β: $\beta = 6$ hours for a sunrise-day sort and $\beta = 10$ hours for a twilight-night sort. From data analysis, we observe that an average of three to four trailers are unloaded at each unloading door during these sorts; therefore, the number of inbound trailers is set to be approximately three to four times the number of unloading doors. Using this information, we create four categories of instances: extra-small (XS), small (S), medium (M), and large (L), as shown in Table 1. Note that the large (L) instance is representative of an instance for a large trucking cross-dock. The XS, S, and M categories were constructed by taking appropriate subsets of the cross-dock doors based on geographic zone information.

Table 1 Instance Types and Parameters

Parameters	XS	S	M	L
Unloading Doors	16	22	33	44
Loading Doors	21	30	44	57
Trailers	$\{50, 64\}$	$\{70, 90\}$	$\{100, 130\}$	$\{130, 180\}$
Sort	{D, N}	{D, N}	{D, N}	{D, N}

The estimated arrival time r_i (in minutes) of every inbound trailer is randomly drawn from the uniform distribution $[0, 60\alpha]$ where $\alpha \in \{0, 2, 4, 6\}$ hours for sunrise-day (D) sort and $\alpha \in \{0, 3, 6, 10\}$ hours for twilight-night (N) sort. Varying the parameter α in this way allows us to analyze the sensitivity of the solution approaches to the spread of the arrival time of trailers during the planning horizon. The number of shipments on inbound trailers depends on the range of average daily shipments anticipated by the terminal managers. The total number of shipments (pallets) on each inbound trailer is drawn from the range $[10, 20]$ and each shipment is randomly assigned to an

outbound loading door. The docking and undocking time of a trailer at a door is set to 5 minutes each.

Outbound trailers in general should be loaded by the end of the sorting period. However, some trailers may have earlier deadlines so that they may be dispatched to meet on-time requirements at downstream terminals. Thus, the outbound trailer deadlines (in minutes) are randomly drawn from a uniform distribution $[60\alpha, 60\beta]$ where $\alpha = 4\ hours, \beta = 6\ hours$ for a sunrise-day sort and $\alpha = 7\ hours, \beta = 10\ hours$ for a twilight-night sort. We set the total number of workers equal to twice the number of unloading doors, and we restrict at most three workers to be used to unload each trailer. We set the value of κ to be 1.25, as recommended in Tadumadze et al. (2019), to calculate the processing time ρ_u^{im}; specifically, we first compute the processing time for a trailer using a single worker and then scale for m workers as given by expressions 1a and 1b. All time parameters in this study such as the trailer arrival times, processing times and loading door deadlines are rounded-up to integer values.

Table 2 summarizes the source of data used to generate the instances for testing the DDD algorithm.

Table 2 Sources of Instances and Parameter Values

Parameters	Real Data	Artificial Data	Comments
Loading and unloading doors	✓		
Distance between doors	✓		
Average forklift speed	✓		
Sort length/Planning horizon	✓		
Number of inbound trailers		✓	Derived from historical data
Trailer arrival times		✓	Historical data shows trailer arrivals are spread in 50-75% of length of planning horizon
Shipment destinations		✓	Randomly generated
Loading deadlines		✓	Derived from sample deadline estimates from research partner
Number of workers		✓	

6.2. Deciding When and How to Generate Feasible Solutions using WDM

In early iterations of the DDD algorithm, it might be unnecessary to solve WDM to find a feasible solution since it may be unlikely to find a solution with a reasonable optimality gap when the time discretization of XDTS-W-LB is relatively coarse. Since the WDM MIP can also be difficult to solve for large instances, we also solve these instances with a time limit and set the Gurobi *MIPFocus* parameter to 1 to prioritize good feasible solutions.

After experimentation, we found that a reasonable approach is to first solve the linear relaxation of WDM yielding objective value W_{LP}. This value is then compared to the XDTS-W-LB optimal

objective function value, LB. If the gap $\frac{W_{LP}-LB}{W_{LP}} \leq 2\%$, the WDM MIP is then solved. We use a MIP time limit of 2 minutes for the XS, S, and M instances and 4 minutes for the L instances. In this way, we direct the algorithm to only spend time searching for good feasible solutions using WDM when the likelihood of identifying a near-optimal solution is higher. Note that for WDM to identify an *optimal* solution to XDTS-W during some iteration, of course $W_{LP} \leq LB$ (thus leading to a computed negative gap).

6.3. Comparison of Exact and Heuristic Approaches

We solve 64 total instance types in this study, 16 for each instance size category (2 sets of trailers, 2 sorts, 4 values of α). Each instance is labeled "ABCE" where A denotes the instance category, B denotes the number of trailers, C denotes the sort and E denotes the value of α. Recall that $\left[0, \frac{60\alpha}{\Delta}\right]$ denotes the *normalized* time window during which trailers arrive at the cross-dock. We generate and solve 10 instances for each instance type, and the run time for each instance is limited to 60 minutes.

We report statistics to compare the performance of exact and heuristic approaches for solving XS (Table 3), S (Table 4), M (Table 5) and L (Table 6) instances of XDTS-W. In these tables, "FullTI" contains statistics for the solution obtained by Gurobi for XDTS-W, "Greedy" contains statistics for the feasible solution obtained by the greedy heuristic, and "DDD" contains statistics for the best solution obtained by the DDD algorithm. Column "$\Theta_{Full}\%$" denotes the average (over 10 instances) optimality gap obtained when solving XDTS-W directly by Gurobi, "T(s)" denotes the average CPU time in seconds, "#" denotes the number of instances (out of 10 instances) solved to optimality, LB_{avg} (or UB_{avg}) denotes the average of the best lower-bound (or upper-bound) values found for the 10 instances in each category. "$\Theta_G\%$" denotes the average percentage gap of the greedy heuristic solution with the best lower-bound obtained by Gurobi. Column "$\Theta\%$" denotes the average optimality gap obtained by the DDD algorithm, where the best lower bound is identified via sequential solution of XDTS-W-LB. Finally, "Iter" reports the average number of iterations to solve the instances by DDD. All optimality gaps are defined as usual as upper bound less lower bound divded by upper bound.

In Tables 3-6, we observe that for the instances in which all trailers arrive at the cross-dock at the start of the planning horizon, i.e., $t = 0$, the optimality gaps are high for both approaches. None of these instances are solved optimally by FullTI, and the average lower bounds computed within the time limit are very weak. It is not surprising that the DDD algorithm performs worse than FullTI in terms of the optimality gap for these instances, given that the approach begins only with time points at $t = 0$ for all doors. Adding time points here is time-consuming, so the DDD algorithm produces weaker average lower bounds than FullTI. However, an interesting observation is that the

Table 3 Statistics for exact and heuristic approaches for instance type XS

Dataset	FullTI					Greedy	DDD					
	Θ_{Full}%	T(s)	#	LB_{avg}	UB_{avg}	Θ_G%	Θ%	T(s)	#	Iter	LB_{avg}	UB_{avg}
XS50D0	73.77	3600	0	33.0	125.9	89.89	76.43	3600	0	5.5	31.3	132.9
XS50D2	0.00	292	10	78.0	78.0	25.36	0.00	133	10	2.9	78.0	78.0
XS50D4	0.00	35	10	310.0	310.0	6.93	0.00	3	10	2.3	310.0	310.0
XS50D6	0.00	26	10	568.6	568.6	3.70	0.00	2	10	1.0	568.6	568.6
XS50N0	73.54	3600	0	33.9	128.1	89.96	76.32	3600	0	5.3	31.5	133.5
XS50N3	0.00	65	10	193.6	193.6	12.40	0.00	14	10	1.2	193.6	193.6
XS50N6	0.00	49	10	517.4	517.4	5.38	0.00	3	10	2.0	517.4	517.4
XS50N10	0.00	58	10	1007.1	1007.1	5.79	0.00	2	10	2.0	1007.1	1007.1
XS64D0	75.10	3600	0	50.5	202.8	88.43	81.77	3600	0	5.3	34.7	200.8
XS64D2	5.98	2266	5	46.4	63.5	43.38	5.27	1988	6	3.3	46.0	62.8
XS64D4	0.00	42	10	257.1	257.1	11.97	0.00	22	10	1.2	257.1	257.1
XS64D6	0.00	35	10	550.5	550.5	6.10	0.00	3	10	1.0	550.5	550.5
XS64N0	75.20	3600	0	50.7	204.6	88.61	82.21	3600	0	5.2	33.8	202.1
XS64N3	0.00	519	10	142.0	142.0	36.88	0.00	144	10	2.2	142.0	142.0
XS64N6	0.00	65	10	510.6	510.6	9.73	0.00	9	10	1.1	510.6	510.6
XS64N10	0.00	55	10	989.9	989.9	8.53	0.00	4	10	1.0	989.9	989.9

upper bounds (and feasible solutions) produced by the DDD algorithm are often much stronger than the ones obtained by FullTI, especially for the M and L instances as shown in Tables 5 and 6. Since it is time-consuming to solve the linear programming relaxation of the larger instances in FullTI, the solver heuristics can only make small improvements in the quality of the feasible solution. On the contrary, the first few iterations in the DDD algorithm are relatively easy to solve due to the small number of decision variables and constraints, and WDM often creates a relatively good primal feasible solution to XDTS-W from the XDTS-W-LB optimal solution.

Figure 5 shows the progression of the bounds generated by the DDD algorithm for an instance of the dataset XS64N0; the initial bounds are very weak, the lower bound improves very slowly as new time points are added in each iteration, and then the DDD algorithm reaches the time limit before it can close the gap further. This instance also shows a particular difficulty; relatively loose trailer loading deadlines lead to optimal solutions to the relaxed problems with zero or very small total deadline delay. Objective functions that discriminate more between different solutions may be more amenable to the DDD algorithm; we will consider problems with tighter deadlines that have this feature in Section 7.

In practical operations, trailers may arrive over time during the planning period. Therefore, we analyze the solution statistics for different values of α; a larger value of *alpha* allows trailer arrivals to be spread over a larger early portion of the planning period. Tables 3-6 demonstrate that as the value of α increases, instances become relatively easier to solve: the average optimality gap and the time taken to solve the instances decreases for all exact and heuristic approaches as the value of α increases. The DDD algorithm solves these easier instances to optimality and faster than

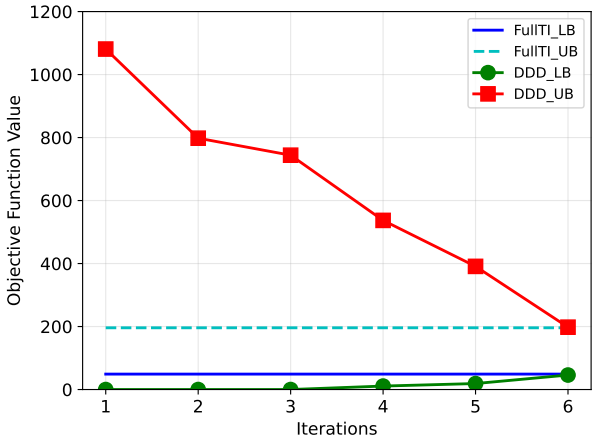


Figure 5 **XS64N0 instance 1: Variation of lower and upper bounds for the DDD algorithm**

Table 4 **Statistics for exact and heuristic approaches for instance type S**

Dataset	FullTI					Greedy	DDD					
	$\Theta_{Full}\%$	T(s)	#	LB_{avg}	UB_{avg}	$\Theta_G\%$	$\Theta\%$	T(s)	#	Iter	LB_{avg}	UB_{avg}
S70D0	71.03	3600	0	94.0	324.5	85.63	79.80	3600	0	5.4	64.0	318.8
S70D2	19.00	2843	3	82.5	104.0	39.19	21.95	2477	4	3.6	81.9	107.7
S70D4	0.00	76	10	399.3	399.3	10.11	0.00	45	10	1.4	399.3	399.3
S70D6	0.00	46	10	737.7	737.7	5.23	0.00	3	10	1.0	737.7	737.7
S70N0	71.13	3600	0	93.8	324.9	85.62	83.87	3600	0	5.1	52.2	324.6
S70N3	0.00	162	10	227.4	227.4	32.98	0.00	81	10	3.0	227.4	227.4
S70N6	0.00	112	10	743.4	743.4	7.42	0.00	10	10	2.0	743.4	743.4
S70N10	0.00	71	10	1448.1	1448.1	3.33	0.00	14	10	2.0	1448.1	1448.1
S90D0	74.36	3600	0	139.9	575.3	84.51	87.81	3600	0	5.1	55.3	453.6
S90D2	69.54	3600	0	78.7	259.2	87.91	74.13	3600	0	4.2	70.2	271.3
S90D4	0.00	116	10	413.8	413.8	14.95	0.00	61	10	2.7	413.8	413.8
S90D6	0.00	59	10	711.8	711.8	5.73	0.00	13	10	2.1	711.8	711.8
S90N0	75.40	3600	0	139.8	616.5	84.54	87.77	3600	0	5.2	55.8	456.4
S90N3	2.25	1325	3	200.3	205.1	42.19	1.98	954	4	3.5	199.6	203.8
S90N6	0.00	203	10	774.6	774.6	4.37	0.00	161	10	1.1	774.6	774.6
S90N10	0.00	93	10	1361.4	1361.4	4.04	0.00	27	10	1.0	1361.4	1361.4

FullTI. Furthermore, we note that the greedy heuristic remains unable to solve these instances to optimality, although the provable optimality gaps drop to $3\% - 10\%$ range.

Table 6 shows that for most large instances, the average upper bounds obtained by DDD are significantly better than the average upper bounds obtained by FullTI. As shown later in Figure 7d, the total number of nodes in the partial network is between 0% to 40% of the total number of nodes in the complete time-expanded network. Therefore, in each iteration of the DDD algorithm, it is relatively easier to solve XDTS-W-LB than XDTS-W directly by a solver.

Table 7 shows the number of instances with better lower and upper bounds from DDD compared to FullTI. We observe that for most of the instances, DDD generates better primal feasible solutions but weaker dual bounds due to the smaller number of time points in the partial network than the

Table 5 Statistics for exact and heuristic approaches for instance type M

Dataset	FullTI					Greedy	DDD					
	Θ_{Full}%	T(s)	#	LB_{avg}	UB_{avg}	Θ_G%	Θ%	T(s)	#	Iter	LB_{avg}	UB_{avg}
M100D0	73.77	3600	0	89.8	342.6	88.17	96.93	3600	0	5.0	9.8	319.4
M100D2	75.76	3600	0	32.7	136.0	93.53	74.88	3600	0	4.1	27.7	114.6
M100D4	0.12	794	9	369.9	370.3	17.08	0.00	277	10	2.8	369.9	369.9
M100D6	0.00	122	10	682.1	682.1	6.69	0.00	38	10	2.0	682.1	682.1
M100N0	73.27	3600	0	90.3	338.2	87.81	96.88	3600	0	5.0	10.1	321.7
M100N3	1.04	1347	7	201.7	203.8	26.24	0.46	991	8	2.4	200.2	202.5
M100N6	0.00	339	10	665.4	665.4	7.01	0.00	131	10	1.0	665.4	665.4
M100N10	0.00	188	10	1400.8	1400.8	4.29	0.00	87	10	1.0	1400.8	1400.8
M130D0	76.60	3600	0	141.7	653.1	86.46	91.90	3600	0	5.3	38.1	471.7
M130D2	80.65	3600	0	57.6	297.6	52.75	95.40	3600	0	5.0	14.4	308.6
M130D4	0.40	983	9	330.0	330.3	37.03	0.00	788	10	2.7	330.0	330.3
M130D6	0.00	150	10	687.3	687.3	10.91	0.00	78	10	1.0	687.3	687.3
M130N0	89.29	3600	0	142.7	2525.3	86.30	91.70	3600	0	5.0	39.0	470.5
M130N3	30.90	3259	2	163.8	244.3	45.33	33.50	3178	4	4.0	159.9	242.8
M130N6	0.00	340	10	663.2	663.2	9.53	0.00	193	10	2.6	663.2	663.2
M130N10	0.00	82	10	1420.2	1420.2	5.00	0.00	12	10	2.2	1420.2	1420.2

Table 6 Statistics for exact and heuristic approaches for instance type L

Dataset	FullTI					Greedy	DDD					
	Θ_{Full}%	T(s)	#	LB_{avg}	UB_{avg}	Θ_G%	Θ%	T(s)	#	Iter	LB_{avg}	UB_{avg}
L130D0	76.34	3600	0	228.3	1218.4	85.09	90.40	3600	0	6.9	72.7	757.6
L130D2	54.53	3600	0	155.7	343.7	64.31	69.00	3600	0	4.3	127.0	409.4
L130D4	0.00	1271	10	747.0	747.0	9.84	0.00	474	10	2.5	747.0	747.0
L130D6	0.00	1163	10	1407.5	1407.5	8.26	0.00	649	10	2.1	1407.5	1407.5
L130N0	78.81	3600	0	228.3	2102.3	84.97	91.40	3600	0	5.9	66.2	773.6
L130N3	0.00	754	10	432.1	432.1	44.02	0.00	398	10	2.5	432.1	432.1
L130N6	0.00	439	10	1383.7	1383.7	5.07	0.00	160	10	1.0	1383.7	1383.7
L130N10	0.00	385	10	2637.7	2637.7	3.38	0.00	140	10	1.0	2637.7	2637.7
L180D0	100.00	3600	0	0.0	3993.3	100.00	100.00	3600	0	6.0	0.0	1554.6
L180D2	82.72	3600	0	313.4	2116.8	66.71	87.80	3600	0	3.4	134.7	1104.5
L180D4	0.00	1005	10	796.7	796.7	39.94	0.00	788	10	3.1	796.7	796.7
L180D6	0.00	318	10	1476.4	1476.4	8.98	0.00	83	10	1.1	1476.4	1476.4
L180N0	100.00	3600	0	0.0	6600.9	100.00	100.00	3600	0	5.0	0.0	1531.4
L180N3	43.08	3600	0	488.5	1268.8	69.09	52.10	3600	0	4.5	425.3	885.7
L180N6	0.00	832	10	1381.0	1381.0	6.20	0.00	290	10	1.3	1381.0	1381.0
L180N10	0.00	499	10	2686.9	2686.9	5.39	0.00	130	10	1.0	2686.9	2686.9

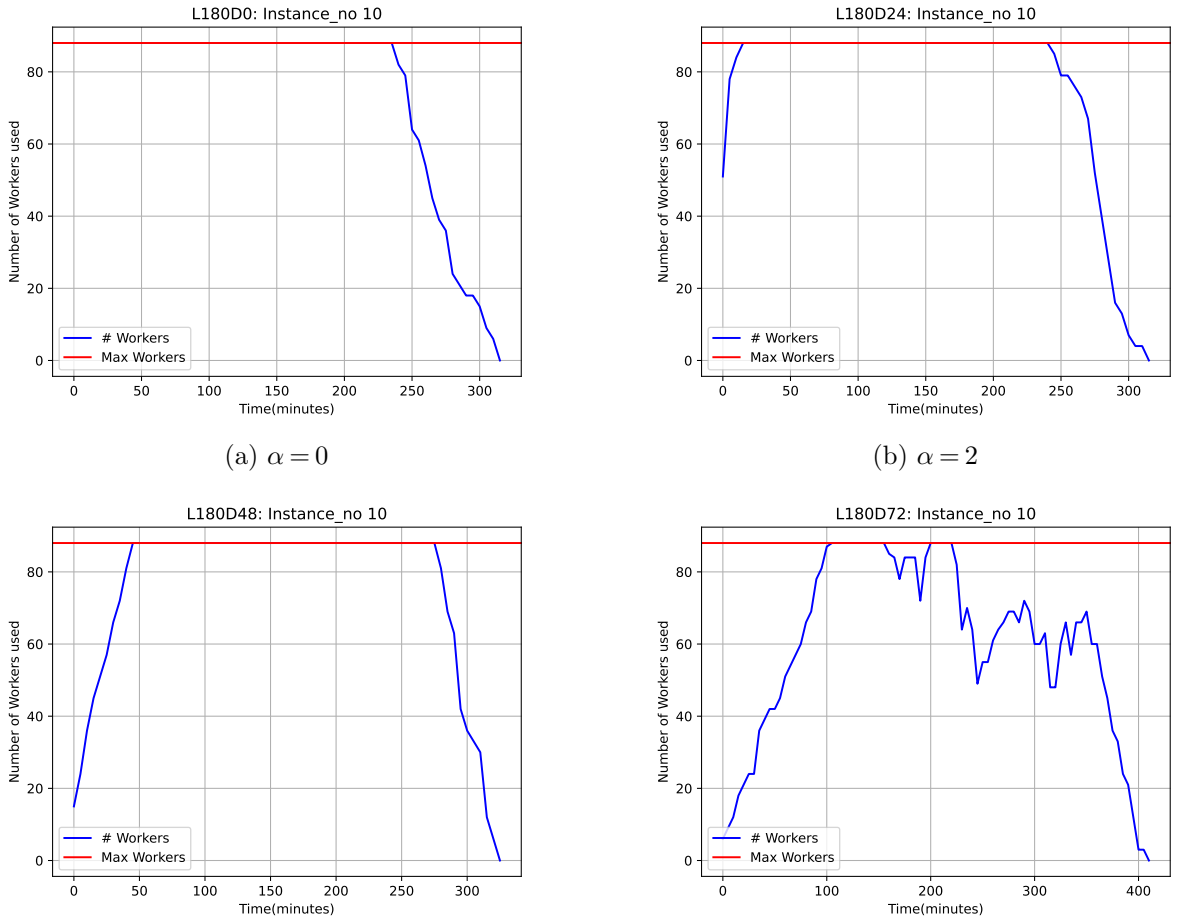
complete time-expanded network. On the other hand, it is interesting to note that for almost half of the instances in each category, the dual bounds produced by DDD are at least as good as the ones obtained from FullTI.

For some instance groups like S70D6, S90N10, and M100N10, the DDD algorithm requires an average number of iterations equal to one, and the average optimality gap is 0%; all of the corresponding 10 instances are solved to optimality in the first iteration. The arrival time window of trailers, in some instances, is wide enough to start unloading each trailer at their arrival times with the available number of workers in the optimal solution. For these instances, DDD is much faster

Table 7 Number of instances (of 160) with better DDD bounds than FullTI bounds

Description	XS	S	M	L
#Instances (out of 160) where DDD UB is *strictly better* than FullTI UB	53	54	84	49
#Instances (out of 160) where DDD UB is *at least as good* as FullTI UB	130	128	144	130
#Instances (out of 160) where DDD LB is *strictly better* than FullTI LB	1	2	2	1
#Instances (out of 160) where DDD LB is *at least as good* as FullTI LB	84	84	65	109

than FullTI because the optimization model size is smaller for DDD due to the smaller number of time points in the partial network.



(a) $\alpha = 0$ (b) $\alpha = 2$

(c) $\alpha = 4$ (d) $\alpha = 6$

Figure 6 Worker usage profiles from greedy heuristic for instance with 44 unloading doors and 88 workers

Increasing the value of α increases the length of the time window during which trailers arrive at the cross-dock. Hence, fewer trailers are required to be unloaded simultaneously at the start of the planning horizon. As a result, some workers remain idle, and the worker constraints are slack for multiple time points in the planning horizon. Figure 6 shows the variation in the number of

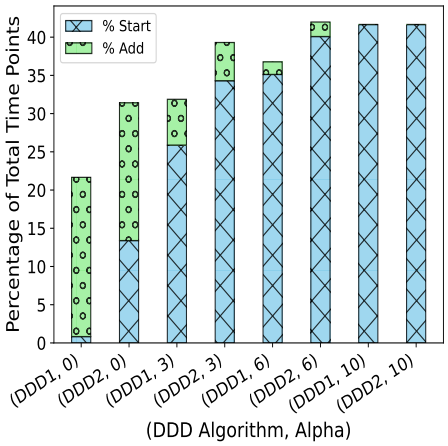
workers busy at any time during the planning horizon for different values of α. The plots correspond to a feasible solution from the greedy heuristic for a large instance with 44 unloading doors and 88 workers. These plots highlight a possible reason why it might be easier for the solver to solve instances with larger values of α for given loading door deadlines. In Figure 6a, all trailers are ready to be unloaded, and the solver needs to prioritize which trailers to unload first and with how many workers; the instances are relatively difficult to solve due to the combinatorial decision space. When $\alpha = 6$ *hours*, as shown in Figure 6d, there are many idle workers at the start of the planning horizon because there are fewer trailers to unload at the start. Therefore, fewer trailer unloading and worker assignment decisions must be made, and most likely, the maximum number of workers (i.e., 3) are assigned to unload each trailer. Note that for these instances with large values of α, the DDD algorithm produces optimal solutions much faster than solving XDTS-W directly by a commercial solver.

6.4. Percentage Time-Space Network Generated By DDD

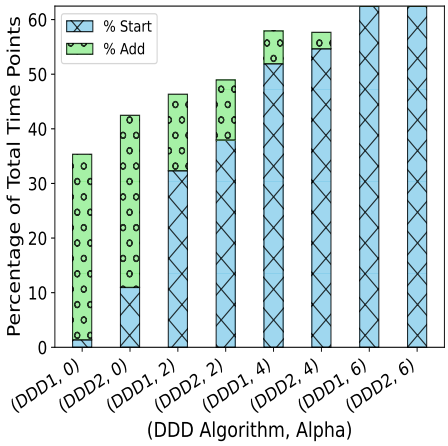
In this section, we conduct experiments to test if different design choices for the initial sets of time points at the unloading doors in the partial network affect the performance of the DDD algorithm. Let DDD1 be the setting where the time points in the partial network are all of the trailer arrival times r_i. In contrast, let DDD2 be the setting where we add a set of uniformly-spaced time points after the latest trailer arrival time: $\left\{(\max_{i\in I} r_i) + \theta\mu : 1 \leq \mu \leq \left\lfloor \frac{T - \max_{i\in I} r_i}{\theta} \right\rfloor, \mu \in \mathbb{Z}^+ \right\}$ where θ is the average of the values in the set $\{\rho_u^{im} : i \ \in I, m \in M, u \in U\}$ rounded to the nearest integer.

Regarding solution quality and computational time, DDD1 always performs better than DDD2. Using too many time points in the partial network for early iterations makes it challenging to solve the corresponding integer programs. As a result, the algorithm reaches the time limit and does not produce better primal and dual bounds compared to FullTI. Figure 7 shows bar plots for DDD1 and DDD2 to compare the *density* of the time-space network in the final iteration of the DDD algorithm. $\%Start$ is defined as the percentage of the total number of time points in the initial partial network, and $\%Add$ is defined as the percentage of the total number of time points added during DDD iterations, relative to the total number of time points across all unloading doors in the complete time-expanded network.

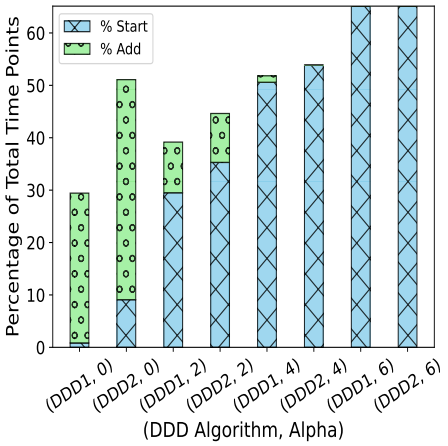
The plots in Figure 7 show that the percentage of time points in the initial set increases, and the percentage of time points added to the partial network decreases with the increase in value of α. As the value of α increases, trailers arrive during a wider time window; therefore, a larger number of time points are included in the initial set of time points in the partial network due to *Property 1*. As a result, many instances with $\alpha = 6$ for D sort and $\alpha = 10$ for N sort solve to optimality in the first iteration, and no new time points are added. For some instances where the XDTS-W-LB



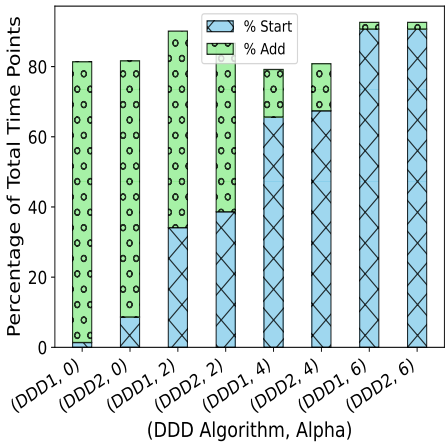
(a) XS instance with 64 trailers and night sort



(b) S instance with 70 trailers and day sort



(c) M instance with 130 trailers and day sort



(d) L instance with 180 trailers and day sort

Figure 7 Percentage network generated by DDD1 and DDD2

optimal solution in the first iteration has a small number of trailers with *short* processing time arcs, adding up to 5% additional time points to the partial network is sufficient to achieve the optimal solution.

7. Computational Study on Instances from Tadumadze et al. (2019)

In this section, we test the DDD algorithm on a related trailer scheduling problem for a different cross-docking setting presented in Tadumadze et al. (2019); instances for this problem are published at the following DOI: 10.5281/zenodo.1487845.. A primary difference in these instances is how the loading door deadlines are computed; the deadline for a door ℓ is determined by assuming that all inbound trailers begin unloading at their arrival times with a single worker and then averaging the completion times of all inbound trailers with shipments for ℓ. Note that in these instances, trailer processing times do not depend on the unloading door u. Since many inbound trailers cannot be unloaded at their arrival times, these instances are more likely to have deadline violations at

many outbound doors when compared to the instances that we used in the computational study in Section 6.

There are some additional differences between XDTS-W and the problem in Tadumadze et al. (2019). First, each inbound trailer has an additional unloading completion time deadline. Second, the objective function is to minimize the total weight of late shipments while XDTS-W penalizes longer lateness by summing total trailer deadline violation. Note that these instances are designed such that a feasible solution always exists; thus, it is possible to compute an upper bound on the optimal objective value without a feasible solution by assuming that each inbound trailer is scheduled with one worker and completed at its unloading deadline. To account for the differences in the two problems we adapt the XDTS-W-LB and WDM MIP models as described in Appendix 9.8.

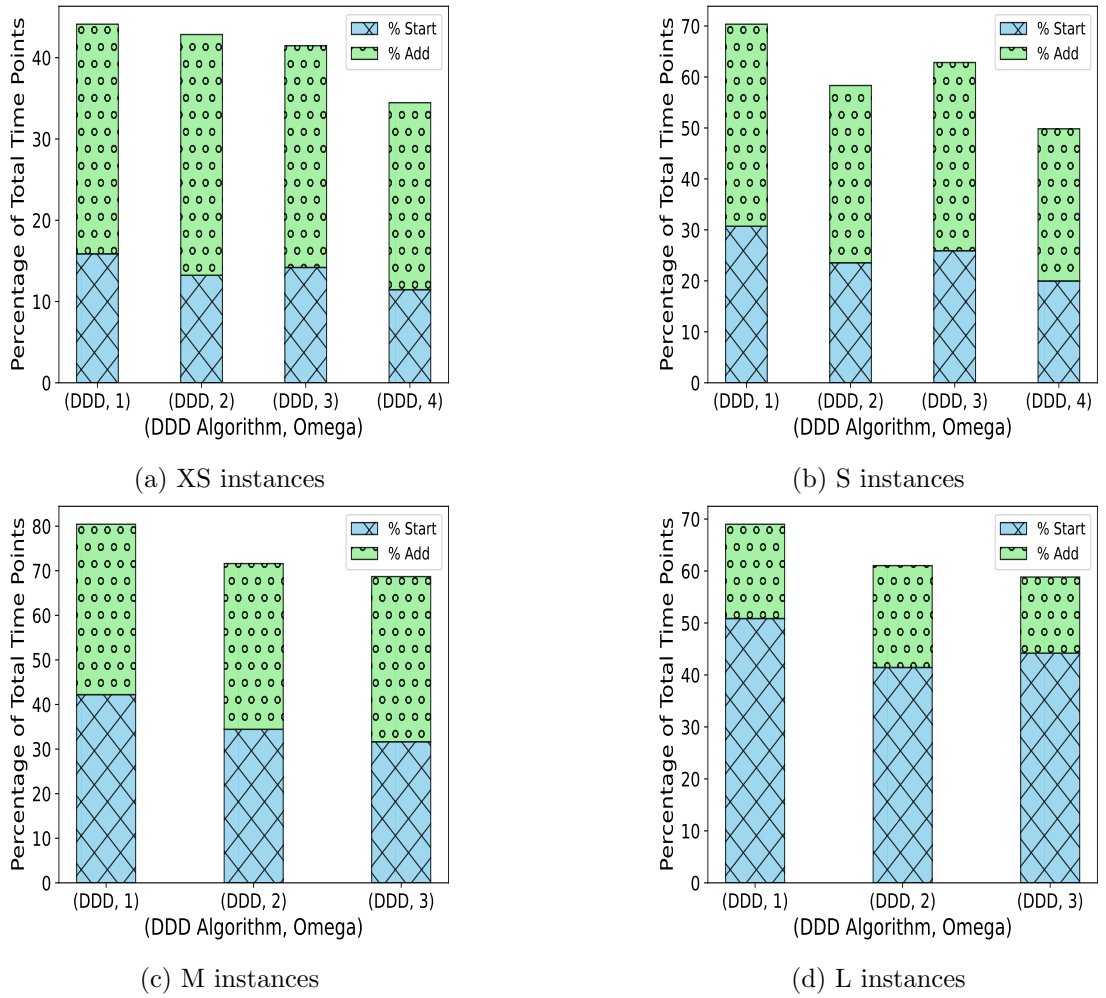
The solution approach proposed in Tadumadze et al. (2019) is to use an interval scheduling approach, as described in Section 2. In the comparative results that follow in this section, we replicate interval scheduling results when $\mu = 1$ which can be shown to be equivalent to solving our adapted XDTS-W formulation with all possible time points. We again refer to this approach with all time points as FullTI.

Table 8 Statistics for exact approaches for instances in Tadumadze et al. (2019)

Dataset	FullTI					DDD					
	Θ_{Full}%	T(s)	#	LB_{avg}	UB_{avg}	Θ%	T(s)	#	Iter	LB_{avg}	UB_{avg}
XS_20_5_1	0.00	0.78	10	441.1	441.1	0.00	0.29	10	4.10	441.1	441.1
XS_20_5_2	0.00	3.97	10	450.5	450.5	0.00	0.72	10	5.00	450.5	450.5
XS_20_5_3	0.00	3.57	10	435.8	435.8	0.00	1.45	10	4.50	435.8	435.8
XS_20_5_4	0.00	5.24	10	430.9	430.9	0.00	0.97	10	4.50	430.9	430.9
S_50_15_1	0.00	33.92	10	870.7	870.7	0.00	28.51	10	4.10	870.7	870.7
S_50_15_2	0.00	51.99	10	871.0	871.0	0.00	42.36	10	4.50	871.0	871.0
S_50_15_3	0.00	59.87	10	836.3	836.3	0.00	49.07	10	4.60	836.3	836.3
S_50_15_4	0.00	96.11	10	849.3	849.3	0.00	122.26	10	6.30	849.3	849.3
M_100_25_1	0.03	442.91	9	1681.9	1682.3	0.00	386.02	10	4.40	1681.9	1681.9
M_100_25_2	0.24	1008.13	7	1622.1	1623.8	0.26	789.36	7	4.70	1620.6	1625.7
M_100_25_3	0.20	1117.97	6	1601.1	1604.2	0.18	1023.68	7	5.10	1600.2	1603.9
L_200_50_1	0.52	1800.00	0	2794.3	2807.0	0.45	1800.00	0	2.78	2794.3	2809.3
L_200_50_2	0.48	1800.00	0	2763.3	2776.6	0.67	1800.00	0	3.00	2763.3	2782.1
L_200_50_3	0.70	1800.00	0	2755.0	2772.5	0.79	1800.00	0	2.60	2755.0	2777.5

Table 8 summarizes the statistics for the performance of our DDD algorithm against solving the instances with the FullTI approach. Each instance is denoted by $A\_B\_C\_D$ where A denotes the instance category, B denotes the number of trailers, C denotes the number of unloading doors and D denotes the value of Ω_{max}. Smaller values of Ω_{max} indicate tight time windows for unloading trailers while larger values provide more flexibility. For the plots in Figure 8, we again define $\%Start$ and $\%Add$ relative to the total number of time points in the complete time-expanded network.

Both approaches solve all XS and S instances to optimality. The DDD algorithm is faster than FullTI for nearly all XS and S instances. The DDD algorithm starts with $10-30\%$ of the complete set of time points on average. It adds approximately 30% new time points to determine a provably optimal solution in $4-5$ iterations on average (see Figures 8a and 8b). However, FullTI is faster than the DDD algorithm for the instances of S_50_15_4; the DDD algorithm executes an average of 6 iterations for the instance. Here, the DDD algorithm must use more iterations to find a solution that can be successfully converted into a provably-optimal solution. One new challenge is that the upper-bounding model (14) is often not able to find a feasible solution due to the unloading deadlines that are not present in XDTS-W. Figure 9b shows the progression of lower and upper bounds for two XS instances, where a feasible solution in the second instance is not identified until iteration 4.



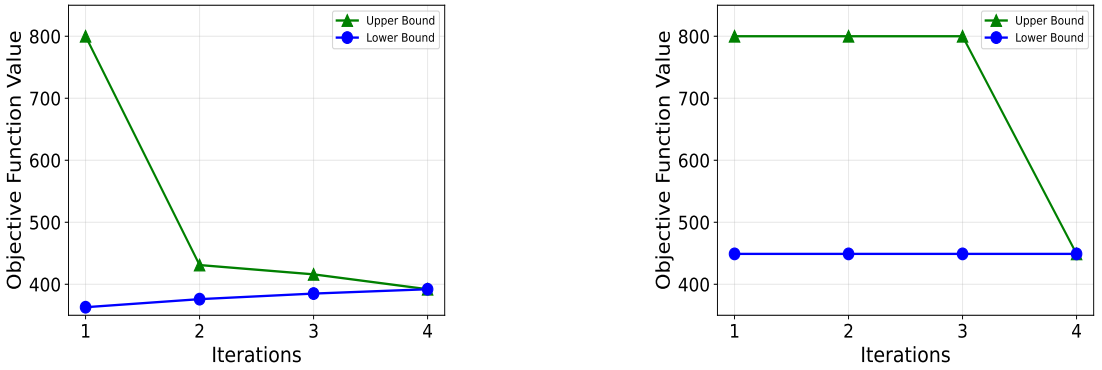
(a) XS instances

(b) S instances

(c) M instances

(d) L instances

Figure 8 Percentage network generated by DDD for instances in Tadumadze et al. (2019)



(a) Instance 1 (b) Instance 2

Figure 9 Lower and Upper bound profiles for two XS instances in Tadumadze et al. (2019)

The DDD algorithm performs similarly to the FullTI approach for the M and L instances, in terms of solution quality, optimality gap, and computation time. Thus, the DDD approach appears to be a robust methodology for many trailer scheduling optimization problems at logistics facilities.

8. Conclusion and Future Directions

In this paper we formulate the cross-dock trailer scheduling problem with constraints on the availability of workers on a complete time-space network. The optimization model prescribes a schedule of trailer unloading activities, including which trailers to unload at which doors in which sequence and with how many workers, to ensure outbound loads can be dispatched with minimal deadline violation. We formulate a MIP model over a partially-expanded network that satisfies certain properties; this MIP model uses careful modeling of the processing time of trailers such that the model is guaranteed to provide a lower bound to the objective value of the model formulated over the complete time-expanded network. We propose an exact algorithm that solves the lower bound MIP model and refines the partial network in each iteration, until a stopping criteria is met. We evaluate our approach on instances derived from actual data representative of an L-shaped cross-dock in Atlanta. For the instances in which trailers arrive during a larger time window, the DDD algorithm is a better alternative to directly solving the instances by a commercial solver in terms of computational time; the DDD algorithm outperforms the greedy heuristic in terms of solution quality. For the instances in which trailers arrive during a small time window, the DDD algorithm often provides better feasible solutions. To establish the effectiveness of our algorithm on instances with different objective functions and additional constraints, we tested the algorithm on instances published by Tadumadze et al. (2019). The DDD algorithm solves most of the small and medium instances faster. For medium and large instances, the DDD algorithm generates solutions and bounds performing similarly to solving with the complete time-expanded network.

The DDD algorithm solves a new lower-bound MIP model in every iteration, even when the lower-bound objective value is already optimal or near-optimal. We are currently exploring re-optimization techniques to use information from the solutions of the lower-bound model in the earlier iterations to warm-start the solver with a feasible solution to the lower-bound model in the current iteration. Any progress in solving the lower-bound models faster can enable the algorithm to solve larger and more complex trailer scheduling instances. Other promising future research directions are the choice of the initial set of time points and the iterative time refinement procedure. The current algorithm adds time points for all trailers with *short* processing time in an iteration. One could test the effectiveness of adding time points for one trailer or a subset of trailers (carefully chosen) in an iteration. It may also be possible to leverage historical trailer scheduling data and use offline learning to build an appropriate partial time-expanded network. Moreover, machine learning-based algorithms or meta-heuristics may be useful to guide the choice of time points to refine the partially time-expanded network.

References

Bartholdi III JJ, Gue KR, 2000 *Reducing labor costs in an ltl crossdocking terminal. Operations research* 48(6):823–832. 2

Boland N, Hewitt M, Marshall L, Savelsbergh M, 2017 *The continuous-time service network design problem. Operations research* 65(5):1303–1321. 6

Boysen N, Briskorn D, Tschöke M, 2013 *Truck scheduling in cross-docking terminals with fixed outbound departures. OR spectrum* 35(2):479–504. 4

Boysen N, Fedtke S, Weidinger F, 2017 *Truck scheduling in the postal service industry. Transportation Science* 51(2):723–736. 5

Boysen N, Fliedner M, 2010 *Cross dock scheduling: Classification, literature review and research agenda. Omega* 38(6):413–422. 4

Buijs P, Vis IF, Carlo HJ, 2014 *Synchronization in cross-docking networks: A research classification and framework. European Journal of Operational Research* 239(3):593–608. 5

Corsten H, Becker F, Salewski H, 2020 *Integrating truck and workforce scheduling in a cross-dock: analysis of different workforce coordination policies. Journal of Business Economics* 90(2):207–237. 5

Edis EB, Oguz C, Ozkarahan I, 2013 *Parallel machine scheduling with additional resources: Notation, classification, models and solution methods. European Journal of Operational Research* 230(3):449–463. 5

Gaudioso M, Monaco MF, Sammarra M, 2021 *A lagrangian heuristics for the truck scheduling problem in multi-door, multi-product cross-docking with constant processing time. Omega* 101:102255. 4

He E, Boland N, Nemhauser G, Savelsbergh M, 2022a *An exact algorithm for the service network design problem with hub capacity constraints. Networks* 80(4):572–596. 6

He EY, Boland N, Nemhauser G, Savelsbergh M, 2022b *Dynamic discretization discovery algorithms for time-dependent shortest path problems. INFORMS Journal on Computing* 34(2):1086–1114. 6

Hermel D, Hasheminia H, Adler N, Fry MJ, 2016 *A solution framework for the multi-mode resource-constrained cross-dock scheduling problem. Omega* 59:157–170. 5

Hewitt M, 2019 *Enhanced dynamic discretization discovery for the continuous time load plan design problem. Transportation Science* 53(6):1731–1750. 6

IIMA, 2021 *9th informs transportation science and logistics society workshop*. `https://conference.iima.ac.in/tsl2021/`. 1

Ladier AL, Alpan G, 2016 *Cross-docking operations: Current research versus industry practice. Omega* 62:145–162. 3, 5

Lagos F, Boland N, Savelsbergh M, 2020 *The continuous-time inventory-routing problem. Transportation Science* 54(2):375–399. 6

Lagos F, Boland N, Savelsbergh M, 2022 *Dynamic discretization discovery for solving the continuous time inventory routing problem with out-and-back routes. Computers & Operations Research* 141:105686. 8

Liao T, Egbelu P, Chang P, 2013 *Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truck schedule in multi-door cross docking operations. International Journal of Production Economics* 141(1):212–229. 4, 5

Marshall L, Boland N, Savelsbergh M, Hewitt M, 2020 *Interval-based dynamic discretization discovery for solving the continuous-time service network design problem. Transportation Science* . 6

Miao Z, Lim A, Ma H, 2009 *Truck dock assignment problem with operational time constraint within cross-docks. European journal of operational research* 192(1):105–115. 5

Scherr YO, Hewitt M, Saavedra BAN, Mattfeld DC, 2020 *Dynamic discretization discovery for the service network design problem with mixed autonomous fleets. Transportation Research Part B: Methodological* 141:164–195. 6

Shabtay D, Steiner G, 2007 *A survey of scheduling with controllable processing times. Discrete Applied Mathematics* 155(13):1643–1666. 5

Shakeri M, Low MYH, Turner SJ, Lee EW, 2012 *A robust two-phase heuristic algorithm for the truck scheduling problem in a resource-constrained crossdock. Computers & Operations Research* 39(11):2564–2577. 5

Sousa JP, Wolsey LA, 1992 *A time indexed formulation of non-preemptive single machine scheduling problems. Mathematical programming* 54(1):353–367. 9

Tadumadze G, Boysen N, Emde S, Weidinger F, 2019 *Integrated truck and workforce scheduling to accelerate the unloading of trucks. European Journal of Operational Research* 278(1):343–362. 3, 4, 5, 6, 7, 20, 27, 28, 29, 30, 39, 40

Tadumadze G, Emde S, Diefenbach H, 2020 *Exact and heuristic algorithms for scheduling jobs with time windows on unrelated parallel machines. OR Spectrum* 42(2):461–497. 4

Van Belle J, Valckenaers P, Cattrysse D, 2012 *Cross-docking: State of the art. Omega* 40(6):827–846. 5

Vu DM, Hewitt M, Boland N, Savelsbergh M, 2020 *Dynamic discretization discovery for solving the time-dependent traveling salesman problem with time windows. Transportation science* 54(3):703–720. 6

Vu DM, Hewitt M, Vu DD, 2022 *Solving the time dependent minimum tour duration and delivery man problems with dynamic discretization discovery. European Journal of Operational Research* 302(3):831–846. 6

9. Appendix

9.1. Proof of Theorem 1

Proof Consider the special case with one loading door with deadline 0, m unloading doors, a set of trailers I, and m workers. Suppose each trailer $i \in I$ has arrival time $r_i = 0$ and that only one worker can be assigned to unload each trailer. In this case, minimizing the deadline violation at the loading door is equivalent to minimizing the makespan of the trailer unloading process at the unloading doors. Therefore, this problem restriction can be used to solve any unrelated parallel machine scheduling problem minimizing makespan, $Rm||C_{max}$, which is known to be strongly NP-Hard; the machines are unrelated because jobs (trailers) can have different processing times on different machines (unloading doors). □

9.2. Proof of Lemma 1

Proof In a feasible solution to XDTS-W, let q trailers start unloading at door u in the sequence $\{1, 2, \cdots, q\}$, with start times $\{\widehat{s}_1, \widehat{s}_2, \cdots, \widehat{s}_q\}$ and completion times $\{\widehat{c}_1, \widehat{c}_2, \cdots, \widehat{c}_q\}$ where $t \leq \widehat{s}_i < t^+ \ \forall \ i \in \{1, 2, \cdots, q\}$. In a complete time-expanded network, (7) is true because trailers have non-zero processing times and constraints (2d) are satisfied in XDTS-W, which implies that at most one trailer is being unloaded at an unloading door at any time.

$$t \leq \widehat{s}_1 < \widehat{c}_1 \leq \widehat{s}_2 < \widehat{c}_2 \leq \cdots < \widehat{c}_{q-1} \leq \widehat{s}_q < t^+ \tag{7}$$

Note that (7) implies that the actual completion time of the trailers $i \in \{1, 2, \cdots, q-1\}$ do not exist in $\mathcal{T}$ in the partial network. From Property 2, the processing time of the trailers $i \in \{1, 2, \cdots, q-1\}$ is 0, and hence, the completion time of the trailers is mapped to time point t in the partial network which gives us $t = s_1 = c_1 = s_2 = c_2 = \cdots = c_{q-1} = s_q$. Therefore at least $q-1$ trailers start and complete unloading at time t in the partial network. □

9.3. Example for Lemma 1

Consider an example with one unloading door (u), one worker and two trailers that have arrival times $r_1 = r_2 = 0$ and processing times $\rho_u^{11} = 1, \rho_u^{21} = 2$. Given a complete time-expanded network with time points $\widehat{\mathcal{T}} = \{0, 1, 2, 3\}$, an optimal solution of XDTS-W can have trailer 1 starting at time $t = 0$ and trailer 2 starting at time $t = 1$ after trailer 1 completes unloading. In a partial network with time points $\mathcal{T} = \{0, 2\}$, one possible solution could have trailer 1 starting at time point $t = 0$ with $\widehat{\rho}_{u0}^{11} = 0$ because the completion time 1 doees not exist in $\mathcal{T}$; trailer 2 starts at time point $t = 0$ with $\widehat{\rho}_{u0}^{21} = 2$. Hence, both trailers start unloading at the same time point $t = 0$, but one of them completes unloading at time $t = 0$.

9.4. Proof of Theorem 2

Proof Consider a feasible solution $(\widehat{x},\widehat{z})$ to XDTS-W defined over a complete time-expanded network $\widehat{G}=(\widehat{N},\widehat{A})$. Now we will construct a feasible solution (x,z) to XDTS-W-LB defined over a partial network $G=(N,A)$ such that the objective function value of solution (x,z) is at most the objective function value of the solution $(\widehat{x},\widehat{z})$. Let $\widehat{\mathcal{T}}$ and $\mathcal{T}$ be the set of time points at any unloading door in $\widehat{G}$ and G, respectively. Recall that the set of time points is same at each unloading door in $\widehat{G}$, by definition, and in G due to Property 1.

From Property 2, for every assignment $\widehat{a}^{im}_{ut}$ to arc $((u,t),(u,t+\rho^{im}_u))$ in $\widehat{G}$ such that $\widehat{x}^{im}_{ut}=1$ for a trailer $i\in I$, there exists an assignment $a^{im}_{ut'}$ to arc $((u,t'),(u,t'+\widehat{\rho}^{im}_{ut'}))$ in G, where $r_i\le t'\le t$ and $\widehat{\rho}^{im}_{ut'}\le\rho^{im}_u$, such that $x^{im}_{ut'}=1$. Therefore, we have

$$\sum_{u\in U}\sum_{\substack{(u,t)\in N\\ t\ge r_i}} x^{im}_{ut}=1,\quad \forall\ i\in I$$
$$x^{im}_{ut}\in\{0,1\},\quad \forall\ i\in I, m\in M,(u,t)\in N, t\ge r_i$$

From Property 2 we know that the completion time of trailers in G is at most the completion time in $\widehat{G}$ and is shown in (8).

$$\sum_{u\in U}\sum_{\substack{(u,t)\in N\\ t\ge r_i}}\left(t+\rho^{im}_u\right)x^{im}_{ut}\le\sum_{u\in U}\sum_{\substack{(u,t)\in \widehat{N}\\ t\ge r_i}}\left(t+\rho^{im}_u\right)\widehat{x}^{im}_{ut} \tag{8}$$

Given deadline violation $\widehat{z}_\ell\ \forall\ \ell\in L$ in the feasible solution to XDTS-W, set $z_\ell=\widehat{z}_\ell\ \forall\ \ell\in L$. Therefore, we have

$$\sum_{u\in U}\sum_{\substack{(u,t)\in N\\ t\ge r_i}}\left(t+\rho^{im}_u\right)x^{im}_{ut}-z_\ell\le\sum_{u\in U}\sum_{\substack{(u,t)\in \widehat{N}\\ t\ge r_i}}\left(t+\rho^{im}_u\right)\widehat{x}^{im}_{ut}-\widehat{z}_\ell\le d_\ell \tag{9}$$

where the second inequality holds from the fact that $(\widehat{x},\widehat{z})$ is a feasible solution to XDTS-W.

Now we will show that the solution x satisfies constraints (10) in XDTW-W-LB; these constraints imply that at most one trailer can be processed at an unloading door at a give time point in G. Recall that constraints (10) are formulated by replacing $\widehat{N}$ with N and $\widehat{\mathcal{A}}_{ut}$ with $\mathcal{A}_{ut}$ for all $(u,t)\in N$ in constraints (2d).

$$\sum_{a^{im}_{ut'}\in\mathcal{A}_{ut}} x^{im}_{ut'}\le 1,\quad \forall\ (u,t)\in N \tag{10}$$

Recall that $\mathcal{A}_{ut}$ at node $(u,t)\in N$ contains only those assignments $a^{im}_{ut'}$ for trailer $i\in I$ and $m\in M$ for which $\widehat{\rho}^{im}_{ut'}>0$ which implies that trailer i occupies unloading door u at time t. Now to prove that solution x satisfies (10) we need to prove that for a given $\widehat{x}$ there is at most one assignment $a^{im}_{ut'}\in\mathcal{A}_{ut}$ such that $x^{im}_{ut'}=1$ and $\widehat{\rho}^{im}_{ut'}>0$ for all $(u,t)\in N$.

Consider the sequence of trailers unloaded at a door u in solution $\widehat{x}$; let (u, m_i, t_i) be the triplet indicating that trailer i is unloaded at door u with m_i workers at the proposed time t_i. For a given time point $t \in \mathcal{T}$ at door u in the partial network G, define $t^+ = \min\{t' : t' > t, t' \in \mathcal{T}\}$ to be the time point immediately next to time t in G. Now given solution $\widehat{x}$ in $\widehat{G}$, we will determine the number of assignments $a^{im}_{ut} \in \mathcal{A}_{ut}$ at node $(u,t) \in N$ in G such that $x^{im}_{ut} = 1$ and $\widehat{\rho}^{im}_{ut} > 0$.

First note that the assignment to processing arcs for the following trailers unloaded at door u in $\widehat{G}$ do not appear in $\mathcal{A}_{ut}$ in constraint (10) for node $(u,t) \in N$.

- trailer i such that $t_i + \rho^{im_i}_u < t^+$. Due to Property 2 the completion time $t_i + \rho^{im_i}_u$ of trailer i is mapped to a time $t' \leq t$ in partial network G such that $t' \in \mathcal{T}$. Therefore, trailer i completes unloading at or before time t.
- trailer i such that $t_i \geq t^+$. Trailer i can start unloading as early as time t^+ in G but not before due to definition (4).

Hence, we only need to consider trailers i with $t_i < t^+ \leq t_i + \rho^{im_i}_u$ unloaded at door u in solution $\widehat{x}$. Note that there can be at most one such trailer assigned to a processing arc active at time t^+ in $\widehat{G}$ because $\widehat{x}$ is feasible to XDTS-W. Therefore, we have at most one assignment in $\mathcal{A}_{ut}$ for which the x-variable can take value 1 and constraint (10) is satisfied for all $(u,t) \in N$. For the sake of completion we mention the two cases below

- If $t_i < t$, then the trailer is assigned to a processing arc $((u,t'_i),(u,t'_i + \widehat{\rho}^{im}_{ut'_i}))$ in G where $r_i \leq t'_i \leq t_i$. The assignment $a^{im}_{ut'_i}$ appears in $\mathcal{A}_{ut}$ in constraints (10) for $(u,t) \in N$ only if $t'_i + \widehat{\rho}^{im}_{ut'_i} \geq t^+$ which implies that $\widehat{\rho}^{im}_{ut'_i} > 0$; in this case, all successive trailers start at or after $t'_i + \widehat{\rho}^{im}_{ut'_i}$.
- If $t \leq t_i < t^+$, then trailer i is assigned to the processing arc $((u,t),(u,t + \widehat{\rho}^{im}_{ut}))$ in G. The assignment a^{im}_{ut} appears in $\mathcal{A}_{ut}$ in constraints (10) for $(u,t) \in N$ only if $t + \widehat{\rho}^{im}_{ut} \geq t^+$ which implies that $\widehat{\rho}^{im}_{ut} > 0$; again all successive trailers start at or after $t + \widehat{\rho}^{im}_{ut}$.

Next we will present a proof by contradiction to show that solution x satisfies the worker constraints (11) in XDTS-W-LB.

$$\sum_{u \in U} \sum_{a^{im}_{ut'} \in \mathcal{A}_{ut}} m x^{im}_{ut'} \leq Q, \quad \forall\ t \in \mathcal{T} \tag{11}$$

The idea is to show that if there exists a time point in $\mathcal{T}$ in the partial network G where constraint (11) is violated, then there must exist a time point $t' \in \widehat{\mathcal{T}}$ in the complete time-expanded network $\widehat{G}$ where constraint (2e) is violated. Let $t \in \mathcal{T}$ be a time point such that $W(t) > Q$ where $W(t)$ is the total number of workers busy unloading trailers at time t. The solid circles in Figure 10 denote the time points in G, and $\widehat{G}$ contains both the solid and hollow circles.

Let $\widehat{s}_i, \widehat{p}_i$ and $\widehat{c}_i\ \forall\ i \in I$ be the start time, processing time and completion time of the unloading process of a trailer i as defined by $\widehat{x}$ in $\widehat{G}$, and let s_i, p_i and c_i be the start time, processing time and completion time of trailer i as defined by x in G.

First, note that $t^+ = \min\{t' : t' > t, t' \in \mathcal{T}\}$ exists in $\widehat{\mathcal{T}}$ because $\mathcal{T} \subset \widehat{\mathcal{T}}$. Let S_1 be the set of trailers that satisfy $s_i \leq \widehat{s}_i < t$ and $t^+ \leq c_i$ for all $i \in S_1$ (see Figure 10); these trailers start before time t and complete at or after time t^+ in both G and $\widehat{G}$. As $t \in \widehat{\mathcal{T}}$, the set of trailers S_1 are being unloaded at time point t in $\widehat{G}$ as well. Let S_2^1 be the set of trailers such that $\widehat{s}_i = s_i = t$ and $t^+ \leq c_i$ for all $i \in S_2^1$; these trailers start at time t and complete at or after time t^+ in both G and $\widehat{G}$. Let S_2^2 be the set of trailers such that $s_i = t$ and $t^+ \leq c_i$ but $t < \widehat{s}_i < t^+$ for all $i \in S_2^2$. Note that $\widehat{s}_i$ gets mapped to an earlier start time s_i in the partial network G due to the definition in (4). We can ignore trailers with $\widehat{s}_i \geq t^+$ as they are not being processed at time t in either G or $\widehat{G}$. We can also ignore trailers with start and completion times equal to t (i.e., $\widehat{\rho} = 0$) because the corresponding terms do not appear in constraint (11).

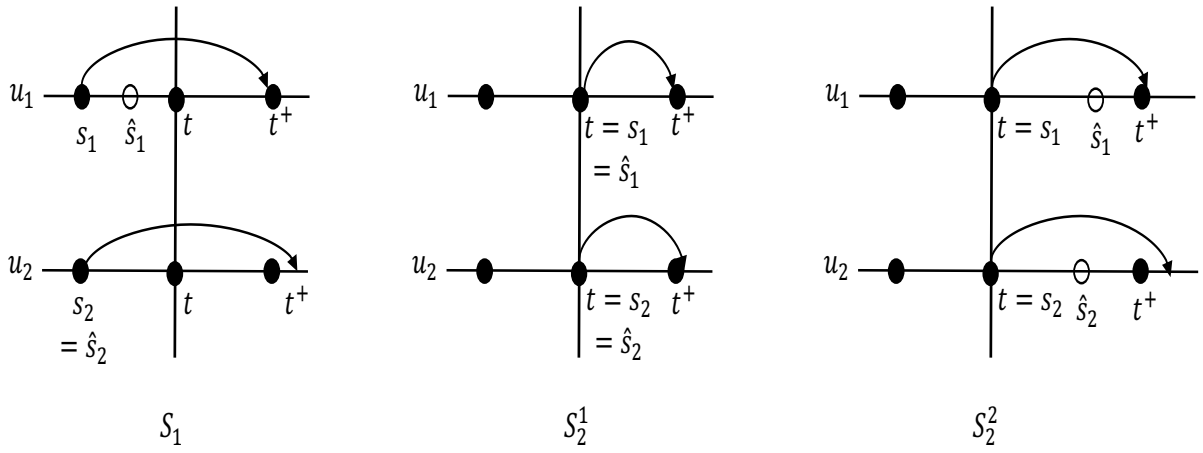
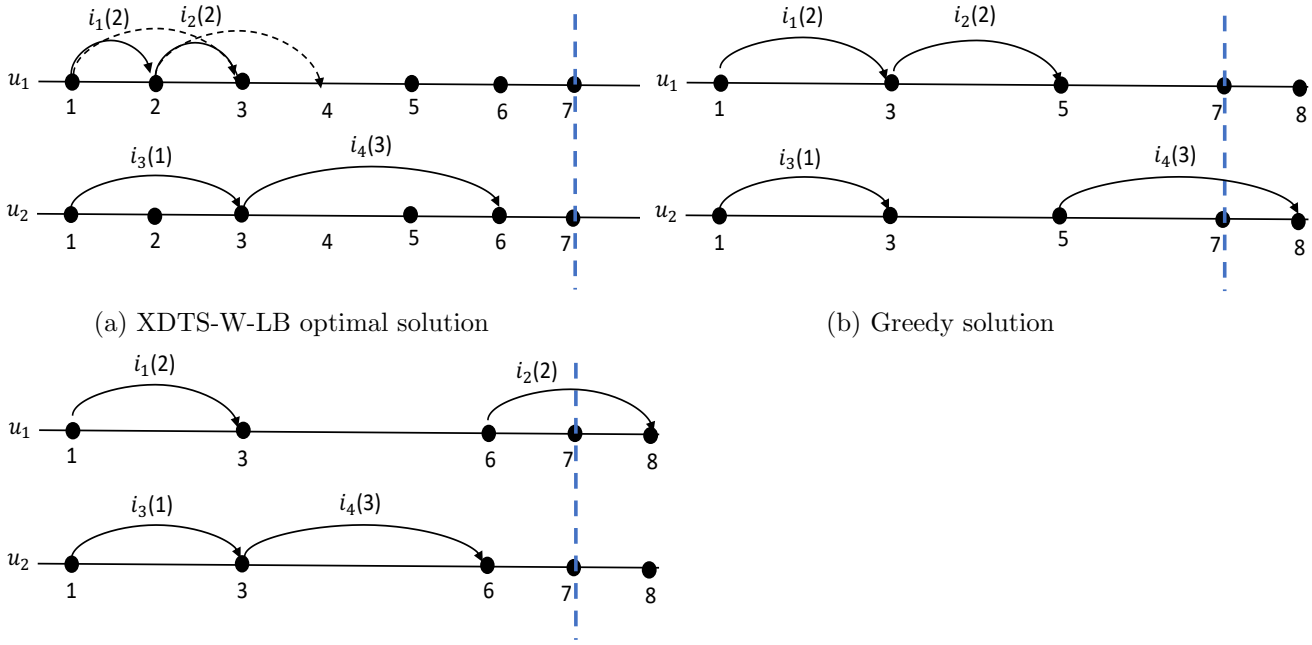


Figure 10 Illustrations for start and completion time of trailers in sets S_1, S_2^1, S_2^2 for Theorem 2

$S_1 \cup S_2^1 \cup S_2^2$ represents the set of trailers that are being unloaded by $W(t)$ workers in the partial network G at time t. Define $\widehat{t} = \min\{t' : t' \geq \widehat{s}_i \ \forall\ i \in S_1 \cup S_2^1 \cup S_2^2\}$. By definition of S_1, S_2^1, S_2^2, $\widehat{t} < t^+$ and all the trailers $i \in S_1 \cup S_2^1 \cup S_2^2$ are busy unloading at time $\widehat{t}$ in $\widehat{G}$ and complete unloading at or after time t^+. Therefore, all the workers busy at time t in G are also busy at time $\widehat{t}$ in $\widehat{G}$. Hence, $W(\widehat{t}) = W(t) > Q$ and constraint (2e) is violated at time $\widehat{t}$, thereby contradicting our initial assumption that $\widehat{x}$ is a feasible solution to XDTS-W. □

9.5. Comparing upper bounds from greedy idea and WDM described in Section 4.2

Consider an example with two unloading doors u_1, u_2, three workers and four trailers i_1, i_2, i_3, i_4. Trailers i_1 and i_2 have a processing time of 2 units at door u_1 and a very large processing time at door u_2. Similarly, trailers i_3 and i_4 have processing times 2 units and 3 units, respectively, at door u_2 and a very large processing time at door u_1. Trailers i_1, i_2, i_3 have shipments destined for one loading door and trailer i_4 has shipments destined for three loading doors. All loading doors have deadline equal to 7 units as indicated by the blue dashed lines in Figure 11.



(a) XDTS-W-LB optimal solution

(b) Greedy solution

(c) WDM solution

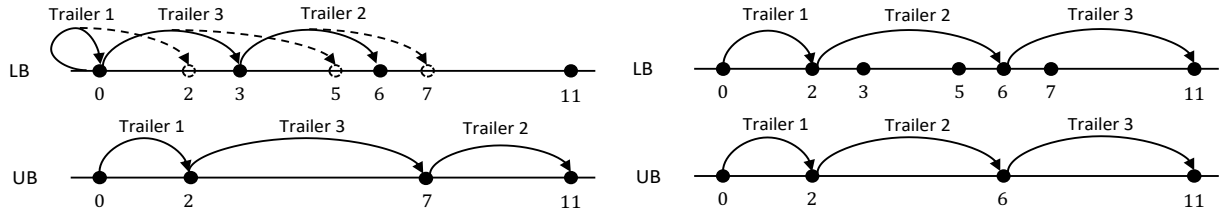
Figure 11 Example to compare upper bounds constructed from XDTS-W-LB optimal solution

Figure 11a illustrates an optimal solution to the XDTS-W-LB with an objective function value of 0 for the given example. The black dashed arcs denote the assignment arcs with actual processing time for trailers i_1 and i_2. The values in the brackets denote the number of workers used to unload the trailer. Figure 11b illustrates a solution based on the greedy idea of delaying the unloading start time of the trailers to construct a feasible solution to XDTS-W from the optimal solution of XDTS-W-LB; the objective value of the solution is $3 \times 1 = 3$ because there is a deadline violation of 1 unit at each of the three loading doors due to trailer i_4. Recall that the greedy approach considers trailers in non-decreasing order of unloading start times determined by the XDTS-W-LB optimal solution, hence, it starts unloading trailer i_2 before trailer i_4. Figure 11c denotes the solution produced by WDM with an objective function value of $1 \times 1 = 1$ because there is a deadline violation of 1 unit at only one loading door due to trailer i_2. This example shows that WDM can produce better upper bounds than the greedy approach.

9.6. Example to show working of the DDD algorithm

Consider the example in Figure 12 with one unloading door, one worker, three inbound trailers that have arrival time $r_1 = r_2 = r_3 = 0$ and actual processing time $p_1 = 2, p_2 = 4, p_3 = 5$ units and initial set of time points $\mathcal{T} = \{0, 3, 6, 10\}$ (indicated by solid dots). Suppose the three trailers contain shipments that need to be cross-docked to two loading doors each with deadline 7 units. In iteration 1, one of the optimal sequence of unloading trailers is $\{1, 3, 2\}$. The hollow dots represent actual completion times of trailers given the start time points. Since these time points are not in $\mathcal{T}$,

the completion times $c_1 = 0, c_3 = 3, c_2 = 6$. The XDTS-W-LB optimal solution is $2 * (7 - 7) = 0$ and the upper bound is $2 * (11 - 7) = 8$. Since, the bounds are unequal, the algorithm adds time points $\{2, 5, 7\}$ to $\mathcal{T}$ which is updated to $\{0, 2, 3, 5, 6, 7, 10\}$. In iteration 2, the XDTS-W-LB optimal solution is equal to the upper bound, i.e 8. Hence, the DDD algorithm terminates. Note that in the final iteration, the partial network has $\frac{7}{12} \times 100 \approx 58\%$ of the total number of start time points in the complete time-expanded network.



(a) Trailer schedule in iteration 1: LB vs UB (b) Trailer schedule in iteration 2: LB vs UB

Figure 12 Solving a toy instance using DDD algorithm

9.7. Proof of Theorem 3

Proof The total number of unique time points in a complete time-expanded network is $T + 1$ $(0, 1, \cdots, T)$. The DDD algorithm starts with a partial network with at least Φ unique time points at every unloading door due to Property 1. In an iteration, if all the trailers are unloaded with their respective actual processing times in XDTS-W-LB optimal solution, then the solution is optimal to XDTS-W because from Theorem 2 XDTS-W-LB optimal objective is a lower bound and all the relaxations (processing time arcs of trailers) are corrected. Hence, no new time points need to be added. If the optimal solution of XDTS-W-LB has trailers with *short* processing time arcs, then the DDD algorithm adds the actual completion time of the trailers at every door in the partial network. Given that nodes are never deleted from the partial network, the DDD algorithm can add at most $(T + 1 - \Phi)$ time points at each unloading door to generate the complete time-expanded network, i.e., $N = \widehat{N}, A = \widehat{A}$. In the worst case, the algorithm would add exactly $|U|$ new time points, one at each unloading door, to the partial network in each iteration. Therefore, the algorithm takes at most $T + 1 - \Phi$ iterations to generate an optimal solution to XDTS-W; in the worst case, the algorithm generates the complete time-expanded network. □

9.8. Modified Lower-bound and Worker Dispatch Models for Tadumadze et al. (2019)

We construct a partial network $G = (N, A)$ based on *Property* 1-3. We define ρ_t^{im} as the processing time of a trailer $i \in I$ that starts unloading with $m \in M$ workers at time $t \in \mathcal{T}$ at any unloading door $u \in U$ in the partial network as shown in (12a); ρ^{im} denotes the actual processing time of trailer

i with m workers and is given in the input data. Let $\mathcal{A}_{ut}$ represent all possible (arc) assignments that occupy unloading door u at time t, that is $\mathcal{A}_{ut} \equiv \{a^{im}_{ut'} : t - \widehat{\rho}^{im}_{t'} < t' \leq t\,,\, ((u,t'),(u,t'+\widehat{\rho}^{im}_{t'})) \in A,\, i \in I,\, m \in M\}$ in G. Equation (12b) defines the weight or cost to start unloading a trailer i at any door u at time point t with m workers where $C^{im} = t + \rho^{im}$ denotes the completion time of a trailer i unloaded by m workers. $\phi_{i\ell}$ denotes the weight of a shipment delivered by trailer i for outbound trailer at door $\ell \in L$ where L is the set of loading doors. $\varphi_{u\ell}$ denotes the time taken to transfer shipment from unloading door u to outbound trailer at loading door ℓ. Let $\bar{d}_i$ be the deadline for unloading each trailer i and d_ℓ be the departure time for outbound trailer from loading door ℓ.

$$\rho^{im}_t = \max\{t' : t \leq t' \leq t + \rho^{im}, t' \in \mathcal{T}\} - t, \quad \forall\, i \in I, m \in M, t \in \mathcal{T} \tag{12a}$$

$$w^{im}_{ut} = \sum_{\ell \in L} \phi_{i\ell} . \begin{cases} 1 \text{ if } C^{im} + \varphi_{u\ell} > d_\ell \\ 0 \text{ otherwise} \end{cases} \quad \forall\, i \in I, m \in M, (u,t) \in N \tag{12b}$$

Based on the above definitions, we adapt the XDTS-W-LB formulation for this problem setup as shown in Model 13.

$$\text{Minimize} \quad \sum_{i \in I} \sum_{m \in M} \sum_{\substack{(u,t) \in N \\ r_i \leq t \leq \bar{d}_i - \rho^{im}}} w^{im}_{ut} x^{im}_{ut} \tag{13a}$$

$$\text{s.t.} \quad \sum_{m \in M} \sum_{\substack{(u,t) \in N \\ r_i \leq t \leq \bar{d}_i - \rho^{im}}} x^{im}_{ut} = 1, \quad \forall\, i \in I \tag{13b}$$

$$\sum_{a^{im}_{ut'} \in \mathcal{A}_{ut}} x^{im}_{ut'} \leq 1, \quad \forall\, (u,t) \in N \tag{13c}$$

$$\sum_{u \in U} \sum_{a^{im}_{ut'} \in \mathcal{A}_{ut}} m x^{im}_{ut'} \leq Q, \quad \forall\, t \in \mathcal{T} \tag{13d}$$

$$x^{im}_{ut} \in \{0,1\} \quad \forall\, i \in I, m \in M, (u,t) \in N, r_i \leq t \leq \bar{d}_i - \rho^{im} \tag{13e}$$

The objective function in (13a) minimizes the total weight of shipments that are late. Constraints (13b) ensure that each trailer is unloaded after it has arrived at the cross-dock and completes unloading before the deadline. Constraints (13c) ensure that at most one trailer is being unloaded at each unloading door at any time point. Constraints (13d) ensure that the worker availability constraints are satisfied at each time point. Constraints (13e) define the domain and range of the variables. Note that we can use the same arguments as shown in the proof of Theorem 3 to show that the DDD algorithm is exact for the problem instances described in Tadumadze et al. (2019).

To formulate the worker dispatch model, consider an optimal solution to Model 13 where (u_i, m_i, t_i) is the triplet indicating that trailer i is unloaded at door u_i with m_i workers at the proposed time t_i. To preserve the unloading order at each door, let parameter $\gamma_{ik} = 1$ if trailer i

is unloaded before trailer k at the same door ($t_i < t_k$ and $u_i = u_k$) and 0 otherwise for all pairs of trailers $i, k \in I$. Also, for ease of notation define a trailer processing time parameter $p_i = \rho^{im_i}$ for each $i \in I$. Integer decision variables $v_{ik} \in \mathbb{Z}_{\geq 0}$ are used in the formulation to determine the number of workers who next unload trailer k after currently unloading trailer i; note that trailer $i = 0$ is used to indicate a dummy source such that v_{0k} counts the number of workers whose first assignment is to trailer k. Binary decision variables y_{ik} must be set to one if at least one worker unloads trailer k after unloading trailer i. Continuous variables $s_i \in \mathbb{R}_{\geq 0}$ denote the unloading start time of trailer $i \in I$. We introduce additional binary variables $h_{i\ell}$ which are set to one if shipments from trailer i are late at loading door ℓ and 0 otherwise for all pairs of $i \in I, \ell \in D_i$. The corresponding worker dispatch model is shown in Model 14 and the objective function of the model minimizes the total weight of shipments that are late.

$$\text{Minimize} \sum_{i \in I} \sum_{\ell \in L} \phi_{il} h_{i\ell} \tag{14a}$$

$$s_i \geq r_i \quad \forall\ i \in I \tag{14b}$$

$$s_i + p_i \leq \bar{d}_i \quad \forall\ i \in I \tag{14c}$$

$$s_i + p_i + \tau_{u_i \ell} - \mathcal{M}^1 h_{i\ell} \leq d_\ell \quad \forall\ i \in I, \ell \in D_i \tag{14d}$$

$$s_i + p_i - \mathcal{M}^2_{ik}(1 - y_{ik}) \leq s_k \quad \forall\ i \in I, k \in \{k' : \gamma_{ik'} \neq 1\} \tag{14e}$$

$$s_i + p_i \leq s_k \quad \forall\ i \in I, k \in \{k' : \gamma_{ik'} = 1\} \tag{14f}$$

$$\sum_{i \in I} v_{0i} \leq Q \tag{14g}$$

$$v_{ki} \leq Q y_{ki} \quad \forall\ i, k \in I \tag{14h}$$

$$\sum_{k \in I \cup \{0\}} v_{ki} \geq \sum_{k \in I} v_{ik} \quad \forall\ i \in I \tag{14i}$$

$$\sum_{k \in I \cup \{0\}} v_{ki} \geq m_i \quad \forall\ i \in I \tag{14j}$$

$$v_{ik} \in \mathbb{Z}_{\geq 0} \quad \forall\ i, k \in I \tag{14k}$$

$$v_{0i} \in \mathbb{Z}_{\geq 0} \quad \forall\ i \in I \tag{14l}$$

$$y_{ik} \in \{0, 1\} \quad \forall\ i, k \in I \tag{14m}$$

$$h_{i\ell} \in \{0, 1\}, \quad \forall\ i \in I, \ell \in D_i \tag{14n}$$

$$s_i \geq 0, \quad \forall\ i \in I \tag{14o}$$

Note that Model 14 has the same set of constraints as shown in WDM (5) except the additional unloading deadline constraints for trailers as shown in (14c), and constraints (14d) which determine if shipments from a trailer are late at a loading door. In constraints (14d) we can set $\mathcal{M}^1 = \max_{i \in I} \bar{d}_i + \max_{u \in U, \ell \in L}\{\tau_{u\ell}\}$. This is true for any $i \in I, \ell \in D_i$ because we have $s_i + p_i \leq \bar{d}_i$, i.e.,

each trailer must be processed by its deadline, and $\tau_{u_i\ell} \leq \max_{u\in U,\ell\in L}\{\tau_{u\ell}\}$. Furthermore, we can set $\mathcal{M}^2_{ik} = \{\bar{d}_i - r_k\}$ as $s_i + p_i \leq \bar{d}_i$ and $s_k \geq r_k \ \forall i \in I, k \in I, k \neq i$.