Planning of Container Crossdocking for an Express Shipment Service Network

Anton J. Kleywegt, Haotian Wu
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332-0205
anton.kleywegt@isye.gatech.edu, haotian.wu@gatech.edu

In air transportation, container crossdocking refers to a loaded container that is transferred at an airport from an incoming flight to an outgoing flight without handling the freight on the container. It reduces handling time and handling cost relative to unloading the container and sorting the freight, and is an economical alternative if a sufficient amount of freight on the incoming flight continues on the outgoing flight. The planning of container crossdocking is very important in express shipment services, both because of its time advantages and its cost advantages. Unfortunately container crossdocking has to be planned before the amount of freight for each origin-destination pair for the day is known. This paper addresses an operational planning problem for an express shipment service network in which flight schedules are given, and container crossdocking decisions have to be made before freight flow demand information is revealed. A two-stage stochastic programming model is formulated in which crossdocking container movements are decided in the first-stage and package flows are determined in the second stage after demand becomes known. The proposed problem takes too long to solve with existing methods, and we address the computational challenges by proposing a two-phase branch-and-Benders-cut approach as well as additional cuts that use characteristics of solutions found so far to improve the progress of the algorithm. We present results from a computational study using real-world data that demonstrates the effectiveness of the proposed approach.

Key words: container crossdocking planning; express shipment service; two-stage stochastic programming; Benders decomposition; level bundle method

1. Introduction

Package delivery represents a significant part of the transportation industry. In China, the total revenue of package delivery has increased from 105.53 billion RMB in 2012 to 1033.23 billion RMB in 2021 (China State Post Bureau 2022). A critical aspect of package delivery is timely service, which is driven, in part, by the growth of e-commerce, which relies heavily on fast delivery. To provide an economically viable delivery service, package express shipment service providers need to carefully allocate and utilize their resources. A primary challenge is to identify consolidation opportunities to keep the costs down while satisfying the service guarantees offered to customers.

An express service provider offers package transportation services to and from specified cities. For each pair of origin city and destination city, the express service provider typically offers a number of different service levels, such as "same day delivery", "next morning delivery", "next day delivery", and "second day delivery". The available service levels may be different for different pairs of origin city and destination city.

An express service provider operates one or more gateway facilities in each served city. The service region of a gateway includes both the origins of some packages and the destinations of some packages. When we consider operations for packages that originate in the service region of a gateway, we refer to the gateway as the origin gateway, and when we consider operations for packages that are destined to the service region of a gateway, we refer to the gateway as the destination gateway. Packages are collected from origin customers, and/or are collected from local stores where origin customers deposited the packages, and are then transported to origin gateways. At the origin gateways the packages are sorted. Sorting includes separation of the packages according to destination gateway and according to service level. In this study, packages are classified by their origin gateway, destination gateway, and service level.

Different package classes may move through the service network in different ways. For example, for a specific pair of origin gateway and destination gateway, packages with a lower service level may be transported from the origin gateway to the destination gateway by ground transportation only, whereas packages with a higher service level may be transported by a combination of ground transportation and air transportation. Packages that travel by a combination of ground transportation and air transportation are moved by ground transportation from their origin gateway to an airport. Different packages can move from the same origin gateway to different airports close to the origin gateway, and different packages can move from different origin gateways to the same airport close to the origin gateways. Such packages then travel with a sequence of one or more connecting flights to airports close to their destination gateways. (The connections involving flights are important, and are discussed below in more detail. Some flights may be operated by the express service provider, and other flights may be operated by other carriers. In the latter case, the express service provider usually enters into long term contracts — covering multiple days — with the other carriers, specifying the amount of capacity on each flight dedicated for the use of the express service provider.) These packages are then moved by ground transportation from the airports to their destination gateways. Similar to the connections between origin gateways and airports, different packages can move from different airports to the same destination gateway close to the airports, and different packages can move from the same airport to different destination gateways close to the airport. At the destination gateways the packages are sorted again, and are then transported via various routes to their destinations.

Air operations constitute a much larger component of the total operational cost of the express delivery service than ground operations. In this study, we focus on the transportation of packages in the carrier's linehaul network that need air transportation to arrive on time. That is, we focus on package flows from origin gateways to destination gateways that use both ground transportation and air transportation, and we do not consider packages that can be delivered on time using ground transportation only, and we do not consider the intra-city operations involving the transportation of packages from origins to origin gateways and from destination gateways to destinations. (For a study of intra-city operations, see, for example, Wu et al. (2022).)

In air transportation, packages are carried on aircraft in containers. Thus, to facilitate air transportation, after packages arrive at an airport from a gateway, the packages are sorted again and are then loaded into containers. Similarly, after packages arrive at the last airport on their journey, the packages are unloaded from the containers, and are then sorted again before their ground transportation from the airport to their destination gateways. Packages can make connections between successive flights (called the inbound flight and the outbound flight) in the following two ways:

1. The packages are unloaded from their containers used on the inbound flight, are then sorted, and then loaded into containers for their outbound flights. Different packages in the same container on the inbound flight do not have to make the same connection. For example, some packages in the container may next travel with ground transportation to their destination gateways, and other packages may next make connections to different outbound flights. This type of connection is called *en-route sorting*.
2. The packages remain in their container used on the inbound flight, and the container is transferred from the aircraft used for the inbound flight to the aircraft used for the outbound flight. This type of connection is called *crossdocking*. The following conditions are necessary to facilitate crossdocking:
   (a) All the packages in the container on the inbound flight must next travel on the same outbound flight on their way to their destination gateways. It is not necessary that all the packages in the container have the same destination gateway. For example, at the end of the outbound flight, the packages may be unloaded from the container, some of these packages may next travel to one destination gateway, some packages may next travel to another destination gateway, and other packages may next make other connections to other flights.
   (b) The same container must fit on both the aircraft used for the inbound flight as well as the aircraft used for the outbound flight. Unlike containers used in ocean shipping (as well as truck and rail transportation), air transportation containers are designed to fit on particular aircraft types. As a result there are various air transportation container types.

An aircraft may be able to carry different container types in different parts of the aircraft, but some container types cannot fit in a given aircraft type. An example of the container configuration that a cargo plane can accommodate is shown in Figure 1. The aircraft can accommodate 8 containers of the same type (numbered $1-8$) and 3 additional containers of other types (labeled X, Y and Z).

All airports can facilitate crossdocking of containers. In contrast, not all airports can facilitate en-route sorting of packages to make connections between inbound flights and outbound flights. Airports that facilitate en-route sorting of packages are sometimes called hubs.

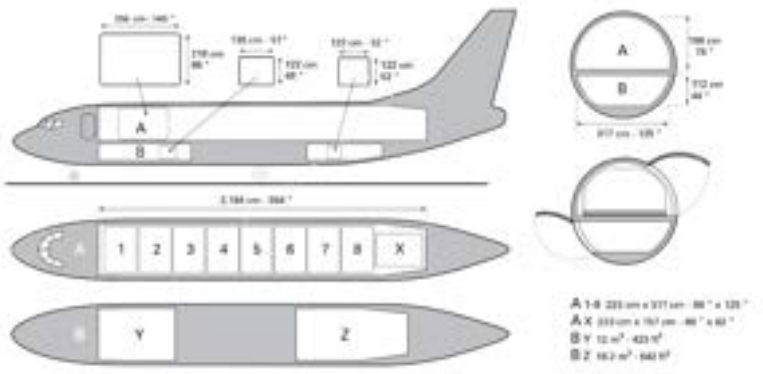


Figure 1 Cargo plane compartment diagram.

Crossdocking of containers have several advantages relative to en-route sorting of packages, including the following:

1. Sorting of packages requires much more equipment, labor, and space than crossdocking of containers. As a result it is much cheaper to crossdock a container than to unload a container, sort the packages, and load the packages into containers.
2. Sorting of packages also requires much more time than crossdocking of containers. As a result crossdocking enables packages to make flight connections that are much too tight to make if the packages have to be unloaded, sorted, and loaded into containers. Making such tight connections may be essential for high service level packages to arrive on time.

Due to the necessary conditions to facilitate crossdocking, mentioned above, careful planning of both container movements and package movements is needed to obtain the benefits of container crossdocking. In this paper we propose an optimization approach to plan container movements and package movements to optimize the benefits of container crossdocking. One reason such planning is challenging is that container movements have to be chosen before the demand for package

transportation becomes known, and there is substantial variation in the daily demand for package transportation. Next we discuss the timing of various decisions relative to the timing of information becoming available.

Due to lengthy administrative processes for approving flight schedules, as well as the need to plan aircraft movements, maintenance schedules, and crew schedules well in advance, carriers choose flight schedules, fleet assignments, and crew schedules weeks or even months before the flights, and often keep flight schedules unchanged for several months. During this period, the package transportation demand can vary from day to day for many reasons, and sometimes such variation can be remarkable. Container movements and crossdocking plans may change at shorter notice than flight schedules, but nevertheless container movements and crossdocking plans for a specific day's operations have to be chosen before that day's demand for package transportation becomes known. In contrast, the movement of individual packages is chosen after most of that day's demand for package transportation becomes known. Therefore, in the proposed optimization problem for planning container movements and package movements, the gateways, airports, flight schedules, and aircraft type used for each flight, have already been determined. For each flight connection in the schedule, the number of containers that make the connection through crossdocking is chosen before the demand for package transportation becomes known. Thereafter the demand is revealed, after which the package flows through the service network is chosen. That is, we formulate a two-stage stochastic program with demand uncertainty, where the number of containers that make each connection through crossdocking are the first-stage decisions, and the package flows are the second stage decisions.

To speed up computation for solving real-world instances, we propose a number of algorithmic strategies. The basic algorithm is a branch-and-Benders-cut algorithm that exploits the decomposition of the second-stage problem into separate problems. Although it is not needed for the branch-and-Benders-cut algorithm, we show that solving the LP relaxation of the problem before starting the branch-and-Benders-cut algorithm helps in two ways. First, it provides a much tighter initial lower bound than could be obtained by other means. Second, we use a level bundle algorithm to solve the LP relaxation, and the linear constraints generated by the level bundle algorithm are used as initial cuts to speed up the branch-and-Benders-cut algorithm. We also show how special cases of the problem based on single scenarios can be solved relatively fast before starting the branch-and-Benders-cut algorithm, and it helps in three ways. First, it provides both a good initial upper bound, as well as a candidate lower bound (although the LP relaxation tends to provide a better lower bound). Second, linear constraints generated at the solutions of the single scenario problems are added to the initial cuts to further speed up the branch-and-Benders-cut algorithm. Third, certain characteristics of the solutions of the single scenario problems are used to generate

additional inequalities that significantly speed up the branch-and-Benders-cut algorithm. As the branch-and-Benders-cut algorithm finds new integer feasible solutions, these additional inequalities are updated to further speed up the algorithm. Computational experiments using data from SF Express demonstrate the effectiveness of the proposed method and also provide insight into the benefits of container crossdocking. We emphasize that although the problems and methods considered in this study are motivated by a collaboration with SF Express, the problem and its data are not an exact representation of operations at SF Express.

The remainder of this paper is organized as follows. In Section 2, we review related prior research. In Section 3, we describe the air container crossdocking planning problem for an express shipment service network. We formulate a two-stage stochastic program with demand uncertainty, and we show how sorting capacity constraints can be expressed efficiently. In Section 4, we introduce a branch-and-Benders-cut algorithm with a first phase designed to improve the efficiency of the second phase, and with added adaptive cuts in the second phase based on characteristics of good solutions found so far. In Section 5, we present and interpret the results of a computational study that used real data.

2. Literature Review

We are not aware of any literature addressing container crossdocking in air transportation, and therefore we briefly review the literature on less-than-truckload (LTL) crossdocking, on air transportation planning for express shipment service, and on Benders decomposition algorithms that have features in common with the proposed solution approach.

2.1. LTL Crossdocking

There is a large research literature on less-than-truckload crossdocking, addressing topics such as service network design, crossdock terminal layout, terminal operations planning, dock assignment, pickup and delivery vehicle routing, and linehaul truck scheduling and dispatching. See, e.g., Agustina et al. (2010), Van Belle et al. (2012), Ladier and Alpan (2016), and Torbali and Alpan (2022) for reviews of the literature on less-than-truckload crossdocking. It is important to point out that although less-than-truckload crossdocking and air transportation crossdocking share the term "crossdocking", the differences between them are crucial for the problem considered in this paper. For example, in air transportation crossdocking the main choice is between (a) dedicating a container to a connection between an incoming flight and an outgoing flight at an airport (in which case the container and all its packages travel on both the incoming flight and the outgoing flight), and (b) unloading the packages from containers used on incoming flights, sorting the packages according to the next leg in their journeys, and loading the packages in containers according to their outgoing flights. The irony is that alternative (a) is called air transportation crossdocking,

but alternative (b) is closer to less-than-truckload crossdocking in terms of operations. Therefore we do not review the literature on less-than-truckload crossdocking in greater detail.

2.2. Air Transportation Planning for Express Shipment Service

There is a substantial literature on planning the air operations for express shipment service. A large part of this literature has considered express shipment service network design problems, which consists of determining a flight network (including flight schedules) that enables the movement of packages from their origins to their destinations at minimum cost in the time specified by the service levels. Some work in this area resulted from collaborations with express service providers. Examples include Kuby and Gray (1993), Armacost et al. (2002), Fleuren et al. (2013), Louwerse et al. (2014), Pérez et al. (2020) and Yıldız and Savelsbergh (2022). As mentioned before, the problem considered in this paper differs from this class of problems in that our problem is addressed after the flight network has been selected, but before daily operational decisions are made, when the number of containers reserved for crossdocking connections between eligible flight pairs are determined while taking demand uncertainty into consideration.

2.3. Benders Decomposition Algorithms

Given the first-stage decisions of a two-stage stochastic program, the second-stage problems separate by scenarios. To solve large-scale two-stage stochastic programs, as in the case of the problem considered in this paper, it is useful to exploit this separation into smaller problems. Benders decomposition, sometimes referred to as a bundle method or L-shaped method in stochastic programming, is such a decomposition technique (see, e.g., Laporte and Louveaux (1993) and Carøe and Tind (1998)). When the first-stage variables are integer and the second-stage variables are continuous, as is the case in the problem considered, the L-shaped method works exactly the same way as Benders decomposition (Bayraksan (2010)). Rahmaniani et al. (2017) present a survey summarizing advantages and drawbacks of the algorithm and addressing different enhancement techniques. Here we briefly mention three major drawbacks that we address in our proposed Benders decomposition based approach. First, instability of first-stage decisions is a widely recognized drawback of the regular Benders decomposition approach, as it results in slow convergence and long computing times due to the excessive oscillations of first-stage decisions (Rubiales et al. (2013)). To stabilize the first-stage decisions, researchers have proposed strategies such as regularized or proximal bundle method, trust-region method, and level bundle method (see, e.g., Ruszczyński (1986), Linderoth and Wright (2003), and Lemaréchal et al. (1995)). However, it has been shown that these stabilization techniques may experience difficulty when applied to optimization problems with discrete first-stage decisions, for which these techniques were not designed (see, e.g., Santoso et al. (2005) and van Ackooij et al. (2015)). A second disadvantage of the approach is that

the upper bounds produced by it often are weak, and problem-specific heuristics are often used to generate better first-stage solutions (see, e.g., Boland et al. (2016), Rahmaniani et al. (2018), and Aboytes-Ojeda et al. (2020)). Third, the regular Benders approach often results in slow improvement of lower bounds, which is especially troublesome if the master problem is a time-consuming discrete optimization problem. The lower bound can have great impact on the size of the branch-and-bound tree generated by the algorithm, and therefore Benders cuts are often generated at fractional nodes of the tree (mostly at the root node at the beginning of the solution process) to rapidly improve the quality of the lower bound. McDaniel and Devine (1977) applied a two-phase strategy, in which the linear relaxation of the master problem is used to quickly improve the lower bound in the first phase, and in the second phase the integrality requirements are reintroduced and the solution process continues. See, e.g., Botton et al. (2013), Naoum-Sawaya and Elhedhli (2013), and Fischetti et al. (2016) for similar two-phase strategies.

3. Problem Description

3.1. Network Description

The express shipment service network has a set G of gateways where packages enter and exit the linehaul network and a set A of airports where flights start and end. In the problem considered here, packages have origin-destination pairs associated with pairs of gateways. Origin-destination gateway pairs are denoted with $(i, j) \in G^2$. The express shipment service offers a set S of service levels, such as "next morning delivery", "next day delivery", etc. Let $G^2_a \subset G^2$ denote the set of origin-destination pairs that use air transportation for some service levels. For some origin-destination pairs $(i, j) \in G^2_a$ only some of the service levels may be available, and only some of these service levels may use air transportation. This is handled by setting the demand, discussed below, equal to zero for combinations of origin-destination pairs and service levels that are not available or do not use air transportation.

Let $A_c \subset A$ denote the set of airports that can support crossdock operations, and let $A_s \subset A$ denote the set of airports that can support en-route sorting between successive flights. Let F denote the set of flights. Each flight $f \in F$ has an origin airport, an origin departure time, a destination airport, a destination arrival time, and various types of capacity constraints. In practice, the weight data of packages for express service usually are more accurate than the volume data. Therefore, package transportation demand, package flows, and capacities of containers and flights, are measured in terms of weight. For each flight $f \in F$, let $U_f \in \mathbb{R}_+$ denote the maximum total weight of packages that flight f can transport. Let C denote the set of container types. For each container type $c \in C$, let $U_c \in \mathbb{R}_+$ denote the maximum total weight of packages that the container type can carry. For each flight $f \in F$, let $C_f \subset C$ denote the set of container types that flight f can accommodate, and for each container type $c \in C_f$, let $M_{f,c} \in \mathbb{N}$ denote the number of containers of type c that flight f can accommodate. Thus, $U_f \leq \sum_{c \in C_f} M_{f,c} U_c$.

3.2. Demand Description

Next we discuss the demand for package transportation. Packages are collected from origin customers, and/or are collected from local stores where origin customers deposited the packages, and are then transported to origin city gateways. Packages arrive at the origin gateways throughout the day. It is important to determine how much demand arrives in time to be transported on each flight. Also, even if packages arrive in time for a flight, due to limited capacity of the flight it may be decided to send the packages on a later flight. Of course, the cumulative amount of demand transported from an origin gateway up to any point in time has to be less than the demand that has arrived at the gateway from its service area up to that point in time, allowing for the time needed for sorting of packages at the gateway. Therefore, first we determine the relevant points in time for measuring the cumulative amounts of demand that have arrived at the origin gateway, by doing the following calculations for each flight $f \in F$. Flight f_5 in the network in Figure 2 will be used to explain the calculations. The departure time of the flight is given. The amount of time needed to process packages departing at the flight's departure airport, such as airport a_2 for flight f_5 (time for unloading packages from a truck, sorting the packages, loading the packages into containers, and loading the containers into the aircraft) is subtracted from the flight departure time, to determine the time at which packages have to arrive at the airport to be in time for the particular flight. Next the ground travel time from each origin gateway that can send packages by ground to the departure airport of the flight is subtracted from the time at which packages have to arrive at the airport, to determine the time at which packages have to leave the gateway to be in time for the flight. In Figure 2, the time at which packages have to leave gateway g_2 and the time at which packages have to leave gateway g_3 to be in time for flight f_5 are determined. It is assumed that sufficient ground transportation capacity is available that all the packages that need ground transportation between a gateway and an airport can be transported at these times. Since ground transportation capacity is much cheaper than air transportation capacity, this is the case in most instances. Next the amount of time needed to process packages at the origin gateways, such as g_2 and g_3 for flight f_5 (time for unloading packages from smaller intra-city trucks, sorting the packages, and loading the packages into larger longhaul trucks) is subtracted from the time at which packages have to leave the origin gateway to be in time for the flight, to determine the time t_{gf} at which the packages have to arrive at the origin gateway g to be in time for flight f. In this way, an origin gateway cutoff time t_{gf} is determined for each flight f and each origin gateway g that can send packages by ground to connect with flight f.

For each origin-destination pair $(i,j) \in G_a^2$, let F_{ij} denote the set of flights f such that origin gateway i can send packages for destination gateway j by ground to connect with flight f. For each destination $j \in G$ and service level $s \in S$, let $F_{j,s}$ denote the set of flights f such that after carried

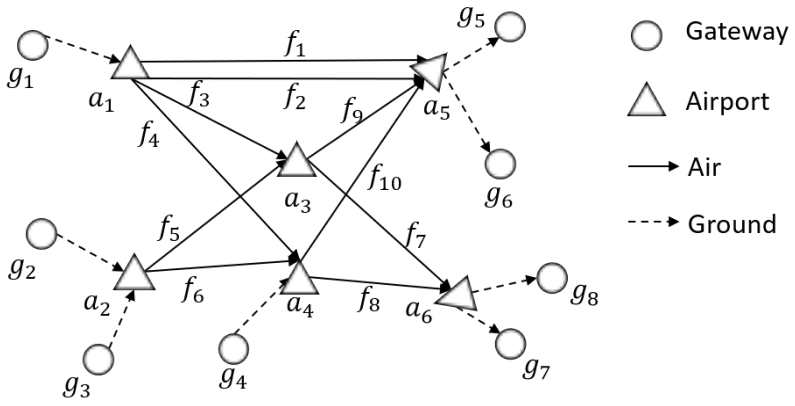


Figure 2 Illustrative partial service network.

by flight f the packages with destination j can be sent by ground transportation from the arrival airport of flight f to gateway j to arrive no later than the cutoff time for service level s. Such cutoff times usually vary among different destination gateways and different service levels, and is earlier than the guaranteed delivery time of the service level at the destination customers, to allow time for sorting at gateways and intra-city transportation. For each origin-destination pair $(i,j) \in G_a^2$ and service level $s \in S$, let $D_{i,j,s,t}$ denote the cumulative demand for package transportation from origin i to destination j with service level s from the beginning of the day until time t. Note that the demand $D_{i,j,s,t}$ does not have to be specified for all times t, but only for cutoff times t_{if} for flights $f \in F_{ij}$. Since demand can vary much from day to day, demand is modeled as a random variable, and $D_{i,j,s,t}$ will also be denoted with $D_{i,j,s,t}(\omega)$ to show its dependence on the scenario ω.

3.3. Decisions

For each airport $a \in A_c$ that can support crossdock operations, let $F_c^2(a)$ denote the set of (ordered) flight pairs $(f_1, f_2) \in F^2$, such that flight f_1 arrives at airport a, flight f_2 departs from airport a, the departure airport of flight f_1 is not equal (or close enough to make ground transportation preferable) to the arrival airport of flight f_2, and containers can make a crossdocking connection from flight f_1 to flight f_2. Making a crossdocking connection from flight f_1 to flight f_2 requires sufficient time between the arrival of flight f_1 and the departure of flight f_2 for the containers to be offloaded from the aircraft used for flight f_1, moved to the aircraft to be used for flight f_2, and loaded onto this aircraft. Making a crossdocking connection from flight f_1 to flight f_2 also requires that there are container types that can be handled by both the aircraft used for flight f_1 and the aircraft used for flight f_2, that is, $C_{f_1} \cap C_{f_2} \neq \varnothing$. Let $F_c^2 := \cup_{a \in A_c} F_c^2(a)$ denote the set of all flight pairs that allow crossdocking connections. For each flight pair $(f_1, f_2) \in F_c^2$, and container type $c \in C_{f_1} \cap C_{f_2}$, let $z_{f_1,f_2,c}$ denote the number of containers of type c that are scheduled to make a crossdocking connection from flight f_1 to flight f_2. This formulation makes provision for

a container to make one crossdocking connection in sequence — two crossdocking connections in sequence can be accommodated with decision variables of the form $z_{f_1,f_2,f_3,c}$, but this is hardly ever needed. These container connections have to be scheduled before the day's demand for package transportation becomes known, and therefore $z_{f_1,f_2,c}$ is a first-stage decision variable in the model.

For each airport $a \in A_s$ that can support en-route sorting operations between successive flights, let $F_s^2(a)$ denote the set of flight pairs $(f_1, f_2) \in F^2$, such that flight f_1 arrives at airport a, flight f_2 departs from airport a, the departure airport of flight f_1 is not equal (or close enough to make ground transportation preferable) to the arrival airport of flight f_2, and packages can make a sorting connection from flight f_1 to flight f_2. Making a sorting connection from flight f_1 to flight f_2 requires sufficient time between the arrival of flight f_1 and the departure of flight f_2 for the containers on flight f_1 to be offloaded, the packages to be unloaded from the containers, sorted, loaded into containers, and the containers to be moved to the aircraft to be used for flight f_2, and loaded on this aircraft. The sorting makes it possible that different packages in the same container on flight f_1 may be put in different containers on flight f_2, or may even connect to different flights and/or different destination gateways after sorting. Therefore it does not matter what container types are used for packages that make sorting connections between flight pairs $(f_1, f_2) \in F_s^2(a)$. Let $F_s^2 := \cup_{a \in A_s} F_s^2(a)$ denote the set of all flight pairs that allow sorting connections. For each flight pair $(f_1, f_2) \in F_c^2$, let $G_c(f_1, f_2) \subset G$ denote the set of destination gateways for which flight pair (f_1, f_2) makes sense. For example, a gateway close to the arrival airport of flight f_2 makes sense, but a gateway close to the departure airport of flight f_1 does not make sense. Similarly, for each flight pair $(f_1, f_2) \in F_s^2$, let $G_s(f_1, f_2) \subset G$ denote the set of destination gateways for which flight pair (f_1, f_2) makes sense.

For each flight pair $(f_1, f_2) \in F_c^2$, each destination gateway $j \in G_c(f_1, f_2)$, and each service level $s \in S$, let $v_{f_1,f_2,j,s}$ denote the amount of demand for destination gateway j and service level s that makes the crossdocking connection from flight f_1 to flight f_2. For each flight pair $(f_1, f_2) \in F_s^2$, each destination gateway $j \in G_s(f_1, f_2)$, and each service level $s \in S$, let $w_{f_1,f_2,j,s}$ denote the amount of demand for destination gateway j and service level s that makes the sorting connection from flight f_1 to flight f_2. For each origin-destination pair $(i, j) \in G_a^2$, each flight $f \in F_{ij}$, and each service level $s \in S$, let $x_{i,j,f,s}$ denote the amount of demand from origin i to destination j with service level s that connects with ground transportation from gateway i to flight f. For each destination $j \in G$, each service level $s \in S$, and each flight $f \in F_{j,s}$, let $y_{j,f,s}$ denote the amount of demand for destination j with service level s that connects with ground transportation from flight f to gateway j. Of course, decisions such as $v_{f_1,f_2,j,s}$, $w_{f_1,f_2,j,s}$, $x_{i,j,f,s}$, and $y_{j,f,s}$ can only be made after the demand becomes known, and therefore $v_{f_1,f_2,j,s}$, $w_{f_1,f_2,j,s}$, $x_{i,j,f,s}$, and $y_{j,f,s}$ are second-stage decision variables in the model. We use notation such as $v_{f_1,f_2,j,s}(\omega)$, $w_{f_1,f_2,j,s}(\omega)$, $x_{i,j,f,s}(\omega)$, and $y_{j,f,s}(\omega)$ when we want to explicitly show the dependence of the decisions on the scenario $\omega \in \Omega$.

3.4. Capacity Constraints

Due to a limited number of forklifts, pallet trucks, conveyor belts, and other equipment, and the constraints imposed by employee schedules, there are constraints on the number of packages that can be sorted at each airport $a \in A_s$ during different time periods. Similarly, there are constraints on the number of containers that can be crossdocked at each airport $a \in A_c$ during different time periods. Next we describe sorting capacity, and the decision variables and constraints that are used to verify that there is sufficient capacity to sort all demand that makes sorting connections during each time period. The purpose of these decision variables and constraints is not to control the sorting process in detail, but just to verify that sorting capacity is sufficient for accommodating timely connections of sorted packages. The decision variables and constraints to verify that crossdocking capacity is sufficient for accommodating timely connections of crossdocking containers will be similar.

Most steps in the sorting of packages that arrive by ground transportation at $a \in A_s$ or depart by ground transportation from $a \in A_s$ take place separately from the en-route sorting of packages connecting between flights. In addition, the timing of ground transportation can easily be adjusted to accommodate the sorting of packages that make connections with ground transportation. Therefore we consider constraints on the capacity to sort packages connecting between flights only. For each airport $a \in A_s$, there are a number of flights in the schedule that arrive at a or depart from a. Let $\{\tau_1(a), \ldots, \tau_{k(a)}(a)\}$ denote the times of flight arrivals and departures at a, ordered such that $\tau_1(a) < \tau_2(a) < \cdots < \tau_{k(a)}(a)$. For each $k \in T(a) := \{1, \ldots, k(a) - 1\}$, let $W_k(a)$ denote the maximum amount of demand that can be sorted at airport a during time interval $[\tau_k(a), \tau_{k+1}(a)]$. For each hub $a \in A_s$, and each pair of flights $(f_1, f_2) \in F_s^2(a)$ that make a sorting connection at a, let $k_1(f_1, f_2) \in \{1, \ldots, k(a) - 1\}$ denote the index of the first time interval during which packages that make a sorting connection from flight f_1 to flight f_2 can be sorted, and let $k_2(f_1, f_2) \in \{1, \ldots, k(a) - 1\}$ denote the index of the last time interval during which packages that make a sorting connection from flight f_1 to flight f_2 can be sorted, with $k_1(f_1, f_2) \leq k_2(f_1, f_2)$. That is, packages that make a sorting connection from flight f_1 to flight f_2 can be sorted during successive time intervals $[\tau_{k_1(f_1,f_2)}(a), \tau_{k_1(f_1,f_2)+1}(a)]$, $\ldots$, $[\tau_{k_2(f_1,f_2)}(a), \tau_{k_2(f_1,f_2)+1}(a)]$. Thus the number of successive time intervals during which packages that make a sorting connection from flight f_1 to flight f_2 can be sorted is $k_2(f_1, f_2) - k_1(f_1, f_2) + 1$.

For each airport $a \in A_s$, each pair of flights $(f_1, f_2) \in F_s^2(a)$ that can make a sorting connection at a, and each $k = k_1(f_1, f_2), \ldots, k_2(f_1, f_2)$, there is a decision variable $u_{f_1,f_2,k}$ that denotes the amount of demand that makes the sorting connection from flight f_1 to flight f_2 and that is sorted in time interval $[\tau_k(a), \tau_{k+1}(a)]$. As for other second-stage decisions, we use notation $u_{f_1,f_2,k}(\omega)$ when we want to explicitly show the dependence of the decisions on the scenario ω.

Similarly, for each airport $a \in A_c$, and each $k \in T(a)$, let $M_k(a)$ denote the maximum number of containers that can be cross-docked at airport a during time interval $[\tau_k(a), \tau_{k+1}(a)]$. For each airport $a \in A_c$, each pair of flights $(f_1, f_2) \in F_c^2(a)$ that can make a crossdocking connection at a, and each $k = k_1(f_1, f_2), \ldots, k_2(f_1, f_2)$, there is a decision variable $\tilde{u}_{f_1,f_2,k}$ that denotes the number of containers that make crossdocking connections from flight f_1 to flight f_2 in time interval $[\tau_k(a), \tau_{k+1}(a)]$.

The objective coefficients for $z_{f_1,f_2,c}$, $v_{f_1,f_2,j,s}$, $w_{f_1,f_2,j,s}$, $x_{i,j,f,s}$, and $y_{j,f,s}$ are denoted $r_{f_1,f_2,c}$, $b_{f_1,f_2,j,s}$, $e_{f_1,f_2,j,s}$, $h_{i,j,f,s}$, and $q_{j,f,s}$ respectively.

3.5. Optimization Problem

Consider a finite set Ω of demand scenarios ω, and let $\pi_\omega \in [0, 1]$ denote the probability of scenario ω. With the decision variables specified above the optimization problem can be written as follows.

$$\min \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c}$$

$$+ \sum_{\omega\in\Omega} \pi_\omega \left[\sum_{(f_1,f_2)\in F_c^2} \sum_{j\in G_c(f_1,f_2)} \sum_{s\in S} b_{f_1,f_2,j,s} v_{f_1,f_2,j,s}(\omega) + \sum_{(f_1,f_2)\in F_s^2} \sum_{j\in G_s(f_1,f_2)} \sum_{s\in S} e_{f_1,f_2,j,s} w_{f_1,f_2,j,s}(\omega) \right. \tag{1a}$$

$$\left. + \sum_{(i,j)\in G_a^2} \sum_{f\in F_{ij}} \sum_{s\in S} h_{i,j,f,s} x_{i,j,f,s}(\omega) + \sum_{j\in G} \sum_{s\in S} \sum_{f\in F_{j,s}} q_{j,f,s} y_{j,f,s}(\omega) \right]$$

$$\text{s.t.} \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, c\in C_{f'}\}} z_{f',f,c} + \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, c\in C_{f'}\}} z_{f,f',c} \;\leq\; M_{f,c} \qquad \forall\, f\in F,\, c\in C_f \tag{1b}$$

$$\sum_{\{i\in G\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, j\in G_c(f',f)\}} v_{f',f,j,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2, j\in G_s(f',f)\}} w_{f',f,j,s}(\omega)$$

$$= \mathbb{1}\{f\in F_{j,s}\} y_{j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, j\in G_c(f,f')\}} v_{f,f',j,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_s^2, j\in G_s(f,f')\}} w_{f,f',j,s}(\omega)$$

$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{1c}$$

$$\sum_{\{i\in G\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2, j\in G_s(f',f)\}} w_{f',f,j,s}(\omega) \;\geq\; \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, j\in G_c(f,f')\}} v_{f,f',j,s}(\omega)$$

$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{1d}$$

$$\sum_{\{f'\in F\,:\,(f',f)\in F_c^2, j\in G_c(f',f)\}} v_{f',f,j,s}(\omega) \;\leq\; \mathbb{1}\{f\in F_{j,s}\} y_{j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_s^2, j\in G_s(f,f')\}} w_{f,f',j,s}(\omega)$$

$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{1e}$$

$$\sum_{j\in G_c(f_1,f_2)} \sum_{s\in S} v_{f_1,f_2,j,s}(\omega) \;\leq\; \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \qquad \forall\,(f_1,f_2)\in F_c^2,\, \omega\in\Omega \tag{1f}$$

$$\sum_{s\in S} \left[\sum_{\{(i,j)\in G^2\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2\}} \sum_{j\in G_s(f',f)} w_{f',f,j,s}(\omega) \right.$$

$$- \sum_{\{f' \in F \,:\, (f,f') \in F_c^2\}} \sum_{j \in G_c(f,f')} v_{f,f',j,s}(\omega) \Bigg]$$

$$\leq \sum_{c \in C_f} U_c \left[M_{f,c} - \sum_{\{f' \in F \,:\, (f',f) \in F_c^2, c \in C_{f'}\}} z_{f',f,c} - \sum_{\{f' \in F \,:\, (f,f') \in F_c^2, c \in C_{f'}\}} z_{f,f',c} \right] \quad \forall\, f \in F,\, \omega \in \Omega \tag{1g}$$

$$\sum_{s \in S} \Bigg[\sum_{\{j \in G \,:\, f \in F_{j,s}\}} y_{j,f,s}(\omega) + \sum_{\{f' \in F \,:\, (f,f') \in F_s^2\}} \sum_{j \in G_s(f,f')} w_{f,f',j,s}(\omega)$$

$$- \sum_{\{f' \in F \,:\, (f',f) \in F_c^2\}} \sum_{j \in G_c(f',f)} v_{f',f,j,s}(\omega) \Bigg]$$

$$\leq \sum_{c \in C_f} U_c \left[M_{f,c} - \sum_{\{f' \in F \,:\, (f',f) \in F_c^2, c \in C_{f'}\}} z_{f',f,c} - \sum_{\{f' \in F \,:\, (f,f') \in F_c^2, c \in C_{f'}\}} z_{f,f',c} \right] \quad \forall\, f \in F,\, \omega \in \Omega \tag{1h}$$

$$\sum_{s \in S} \Bigg[\sum_{\{(i,j) \in G^2 \,:\, f \in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f' \in F \,:\, (f',f) \in F_c^2\}} \sum_{j \in G_c(f',f)} v_{f',f,j,s}(\omega)$$

$$+ \sum_{\{f' \in F \,:\, (f',f) \in F_s^2\}} \sum_{j \in G_s(f',f)} w_{f',f,j,s}(\omega) \Bigg] \leq U_f \qquad \forall\, f \in F,\, \omega \in \Omega \tag{1i}$$

$$\sum_{\{f' \in F_{ij} \,:\, t_{if'} \leq t_{if}\}} x_{i,j,f',s}(\omega) \leq D_{i,j,s,t_{if}}(\omega) \qquad \forall\, (i,j) \in G_a^2,\, s \in S,\, f \in F_{ij},\, \omega \in \Omega \tag{1j}$$

$$\sum_{\left\{(f_1,f_2) \in F_c^2(a) \,:\, k_1(f_1,f_2) \leq k \leq k_2(f_1,f_2)\right\}} \tilde{u}_{f_1,f_2,k} \leq M_k(a) \qquad \forall\, a \in A_c,\, k \in T(a) \tag{1k}$$

$$\sum_{k=k_1(f_1,f_2)}^{k_2(f_1,f_2)} \tilde{u}_{f_1,f_2,k} = \sum_{c \in C_{f_1} \cap C_{f_2}} z_{f_1,f_2,c} \qquad \forall\, (f_1,f_2) \in F_c^2 \tag{1l}$$

$$\sum_{\left\{(f_1,f_2) \in F_s^2(a) \,:\, k_1(f_1,f_2) \leq k \leq k_2(f_1,f_2)\right\}} u_{f_1,f_2,k}(\omega) \leq W_k(a) \qquad \forall\, a \in A_s,\, k \in T(a),\, \omega \in \Omega \tag{1m}$$

$$\sum_{k=k_1(f_1,f_2)}^{k_2(f_1,f_2)} u_{f_1,f_2,k}(\omega) = \sum_{j \in G_s(f_1,f_2)} \sum_{s \in S} w_{f_1,f_2,j,s}(\omega) \qquad \forall\, a \in A_s,\, (f_1,f_2) \in F_s^2(a),\, \omega \in \Omega \tag{1n}$$

$$\tilde{u}_{f_1,f_2,k} \in \mathbb{R}_+ \qquad \forall\, a \in A_c,\, (f_1,f_2) \in F_c^2(a),\, k \in \{k_1(f_1,f_2), \ldots, k_2(f_1,f_2)\} \tag{1o}$$

$$u_{f_1,f_2,k}(\omega) \in \mathbb{R}_+ \qquad \forall\, a \in A_s,\, (f_1,f_2) \in F_s^2(a),\, k \in \{k_1(f_1,f_2), \ldots, k_2(f_1,f_2)\},\, \omega \in \Omega \tag{1p}$$

$$v_{f_1,f_2,j,s}(\omega) \in \mathbb{R}_+ \qquad \forall\, (f_1,f_2) \in F_c^2,\, j \in G_c(f_1,f_2),\, s \in S,\, \omega \in \Omega \tag{1q}$$

$$w_{f_1,f_2,j,s}(\omega) \in \mathbb{R}_+ \qquad \forall\, (f_1,f_2) \in F_s^2,\, j \in G_s(f_1,f_2),\, s \in S,\, \omega \in \Omega \tag{1r}$$

$$x_{i,j,f,s}(\omega) \in \mathbb{R}_+ \qquad \forall\, (i,j) \in G_a^2,\, f \in F_{ij},\, s \in S,\, \omega \in \Omega \tag{1s}$$

$$y_{j,f,s}(\omega) \in \mathbb{R}_+ \qquad \forall j \in G,\, s \in S,\, f \in F_{j,s},\, \omega \in \Omega \tag{1t}$$

$$z_{f_1,f_2,c} \in \mathbb{N} \qquad \forall (f_1,f_2) \in F_c^2,\, c \in C_{f_1} \cap C_{f_2}. \tag{1u}$$

Constraints (1b) limit the number of crossdocking containers of different types carried by each flight. Constraints (1c) enforce the flow balance on each flight for each service level in every scenario. Constraints (1d) and (1e) ensure sufficient demand to be crossdocked on each flight in every scenario. Constraints (1f) are container capacity constraints limiting the weight of crossdocking demands accommodated by exchanged containers on each pair of flights in every scenario. Constraints (1g) and (1h) limit the weight of demands accommodated by non-crossdocking containers on each flight in every scenario. Constraints (1i) are flight capacity constraints limiting the weight of demands accommodated by each flight in every scenario. Constraints (1j) ensure that served demand for each service level does not exceed amount available in every scenario. Constraints (1k) and (1l) enforce the crossdocking capacity at each airport. Constraints (1m) and (1n) enforce the sorting capacity at each airport in every scenario. Constraints (1q)–(1t) define the flow variables and their domains. Constraints (1u) enforce integer number of containers exchanged by each pair of eligible flights. Note that constraints (1d) and (1c) imply (1e), and constraints (1e) and (1c) imply (1d). Thus constraints (1d) or (1e) can be omitted. Similarly, constraints (1c) and (1g) imply (1h), and constraints (1c) and (1h) imply (1g). Thus constraints (1g) or (1h) can be omitted.

3.6. Alternative Formulation of Capacity Constraints

Next we show that we can eliminate decision variables $\tilde{u}_{f_1,f_2,k}$ and $u_{f_1,f_2,k}$, and replace constraints (1k)–(1p) with the following constraints:

$$\sum_{\left\{(f_1,f_2)\in F_c^2(a)\,:\,k_1\le k_1(f_1,f_2)\le k_2(f_1,f_2)\le k_2\right\}} \sum_{c\in C_{f_1}\cap C_{f_2}} z_{f_1,f_2,c} \;\le\; \sum_{k=k_1}^{k_2} M_k(a)$$
$$\forall\, a \in A_c,\, k_1,k_2 \in T(a),\, k_1 \le k_2 \tag{2}$$

$$\sum_{\left\{(f_1,f_2)\in F_s^2(a)\,:\,k_1\le k_1(f_1,f_2)\le k_2(f_1,f_2)\le k_2\right\}} \sum_{j\in G_s(f_1,f_2)} \sum_{s\in S} w_{f_1,f_2,j,s}(\omega) \;\le\; \sum_{k=k_1}^{k_2} W_k(a)$$
$$\forall\, a \in A_s,\, k_1,k_2 \in T(a),\, k_1 \le k_2,\, \omega \in \Omega \tag{3}$$

To show this, consider the following transportation problem with simpler notation. Let S and T be two finite sets, and let $A \subset S \times T$ denote the set of ordered pairs $(i,j) \in S \times T$ that can be assigned to each other. For each $i \in S$, let $\delta_i^+ := \{j \in T : (i,j) \in A\}$ denote the set of all $j \in T$ that can be assigned to i, and for each $j \in T$, let $\delta_j^- := \{i \in S : (i,j) \in A\}$ denote the set of all $i \in S$ that

can be assigned to j. For each $i \in S$, let $a_i \geq 0$ denote the "supply" at i, and let $a := (a_i,\, i \in S)$. For each $j \in T$, let $b_j \geq 0$ denote the "demand" at j, and let $b := (b_j,\, j \in T)$. For any $a \geq 0$, $b \geq 0$, let

$$X(a,b) \;:=\; \left\{ x = (x_{i,j},\, (i,j) \in E) \,:\, \sum_{j \in \delta_i^+} x_{ij} \leq a_i \;\forall\; i \in S,\; \sum_{i \in \delta_j^-} x_{ij} = b_j \;\forall\; j \in T,\; x_{ij} \geq 0 \;\forall\; (i,j) \in A \right\}$$

denote the set of all feasible "flows" between supplies a and demands b. For any $a \geq 0$, let

$$B(a) \;:=\; \{ b = (b_j,\, j \in T) \,:\, b \geq 0,\, X(a,b) \neq \varnothing \}$$

denote the set of all feasible demands for supply a. Let

$$Y(a) \;:=\; \left\{ (b,x) \,:\, b \geq 0,\, \sum_{j \in \delta_i^+} x_{ij} \leq a_i \;\forall\; i \in S,\; \sum_{i \in \delta_j^-} x_{ij} = b_j \;\forall\; j \in T,\; x_{ij} \geq 0 \;\forall\; (i,j) \in A \right\}.$$

Note that $B(a) = \{ b \geq 0 : \exists\, x \text{ s.t. } (b,x) \in Y(a) \}$ is the projection of the polytope $Y(a)$ onto the b-subspace. Therefore $B(a)$ is a polytope. For any $U \subset T$, $U \neq \varnothing$, and any $a \geq 0$, let

$$H^U(a) \;:=\; \left\{ b = (b_j,\, j \in T) \,:\, \sum_{j \in U} b_j = \sum_{i \,\in\, \bigcup_{j \in U} \delta_j^-} a_i \right\}$$

denote a hyperplane in the b-subspace, and let

$$H(a) \;:=\; \left\{ b = (b_j,\, j \in T) \,:\, b \geq 0,\, \sum_{j \in U} b_j \leq \sum_{i \,\in\, \bigcup_{j \in U} \delta_j^-} a_i \;\forall\; U \subset T, U \neq \varnothing \right\}$$

denote a polytope in the b-subspace.

Proposition 1. *For any vector of supplies $a := (a_i,\, i \in S) \geq 0$, it holds that $B(a) = H(a)$.*

The proof of Proposition 1 is given in Appendix A.1.

Next we show by example that the number of facets of $B(a)$ may be exponential in $|T|$, and then we show that the structure of δ_j^- in our problem allows $B(a)$ to be given by a small number of inequalities.

Example 1. For $n \in \{1, 2, \ldots\}$, let $S = \{0, 1, \ldots, n\}$ and $T = \{1, \ldots, n\}$. For each $j \in T$, let $\delta_j^- = \{0, j\}$. That is, $\delta_0^+ = T$ and $\delta_i^+ = \{i\}$ for $i \in \{1, \ldots, n\}$. Let $a > 0$. It follows that for any $U \subset T$, $U \neq \varnothing$,

$$H^U(a) \;=\; \left\{ b = (b_j,\, j \in T) \,:\, \sum_{j \in U} b_j = a_0 + \sum_{i \in U} a_i \right\}$$

and

$$H(a) \;=\; \left\{ b=(b_j,\, j\in T) \,:\, b\geq 0,\, \sum_{j\in U} b_j \leq a_0 + \sum_{i\in U} a_i \;\forall\; U\subset T, U\neq\varnothing \right\}.$$

For every $U\subset T$, $U\neq\varnothing$, and $j\in U$, let

$$b_j^U \;:=\; \begin{cases} \frac{a_0}{|U|}+a_i & \text{if } j\in U \\ 0 & \text{otherwise} \end{cases}$$

and $x^U_{0,j} := a_0/|U|$, $x^U_{j,j} := a_j$ for all $j\in U$; and $x^U_{i,j}=0$ otherwise. Note that $b^U\geq 0$ and $x^U\in X(a,b^U)$, and thus $b^U\in B(a)$. Also note that $b^U\in H^U(a)$.

Next we show that $b^U\notin H^{U'}(a)$ for all $U,U'\subset T$, $U,U'\neq\varnothing$, $U\neq U'$.

Case 1: There is an $\hat{j}\in U\setminus U'$.

Then

$$\sum_{j\in U'} b_j^U \;=\; \sum_{j\in U\cap U'} b_j^U \;\leq\; \sum_{j\in U\setminus\{\hat{j}\}} \frac{a_0}{|U|} + \sum_{j\in U\cap U'} a_j \;<\; a_0 + \sum_{j\in U'} a_j$$

and thus $b^U\notin H^{U'}(a)$.

Case 2: There is an $\hat{j}\in U'\setminus U$.

Then

$$\sum_{j\in U'} b_j^U \;=\; \sum_{j\in U\cap U'} b_j^U \;\leq\; \sum_{j\in U\cap U'} \frac{a_0}{|U|} + \sum_{j\in U'\setminus\{\hat{j}\}} a_j \;<\; a_0 + \sum_{j\in U'} a_j$$

and thus $b^U\notin H^{U'}(a)$. Hence $b^U\in B(a)\cap H^U(a)\setminus H^{U'}(a)$ for all $U,U'\subset T$, $U,U'\neq\varnothing$, $U\neq U'$, and therefore $B(a)\cap H^U(a)$ is a facet of $B(a)$ for each $U\subset T$, $U\neq\varnothing$. Thus $B(a)$ has at least $2^{|T|}-1$ facets. □

Proposition 2. *Suppose that $S=\{1,\ldots,K\}$, and that for each $j\in T$ it holds that $\delta_j^- = \{k_1(j),\ldots,k_2(j)\}$ for some $1\leq k_1(j)\leq k_2(j)\leq K$. For any $a\geq 0$, let*

$$P(a) \;:=\; \left\{ b=(b_j,\, j\in T) \,:\, b\geq 0,\, \sum_{\{j\in T\,:\,k_1\leq k_1(j)\leq k_2(j)\leq k_2\}} b_j \leq \sum_{i=k_1}^{k_2} a_i \;\forall\; 1\leq k_1\leq k_2\leq K \right\}.$$

Then $H(a)=P(a)$.

The proof of Proposition 2 is given in Appendix A.2.

It follows that problem (1) can be solved by solving problem (4):

$$\begin{aligned} \min \quad & \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c} \\ & + \sum_{\omega\in\Omega} \pi_\omega \left[\sum_{(f_1,f_2)\in F_c^2} \sum_{j\in G_c(f_1,f_2)} \sum_{s\in S} b_{f_1,f_2,j,s} v_{f_1,f_2,j,s}(\omega) + \sum_{(f_1,f_2)\in F_s^2} \sum_{j\in G_s(f_1,f_2)} \sum_{s\in S} e_{f_1,f_2,j,s} w_{f_1,f_2,j,s}(\omega) \right. \end{aligned} \tag{4a}$$

$$+ \sum_{(i,j)\in G_a^2} \sum_{f\in F_{ij}} \sum_{s\in S} h_{i,j,f,s} x_{i,j,f,s}(\omega) + \sum_{j\in G} \sum_{s\in S} \sum_{f\in F_{j,s}} q_{j,f,s} y_{j,f,s}(\omega) \Bigg]$$

$$\text{s.t.} \quad \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, c\in C_{f'}\}} z_{f',f,c} + \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, c\in C_{f'}\}} z_{f,f',c} \;\leq\; M_{f,c} \qquad \forall\, f\in F,\, c\in C_f \tag{4b}$$

$$\sum_{\{i\in G\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, j\in G_c(f',f)\}} v_{f',f,j,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2, j\in G_s(f',f)\}} w_{f',f,j,s}(\omega)$$
$$= \; \mathbb{1}\{f\in F_{j,s}\} y_{j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, j\in G_c(f,f')\}} v_{f,f',j,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_s^2, j\in G_s(f,f')\}} w_{f,f',j,s}(\omega)$$
$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{4c}$$

$$\sum_{\{i\in G\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2, j\in G_s(f',f)\}} w_{f',f,j,s}(\omega) \;\geq\; \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, j\in G_c(f,f')\}} v_{f,f',j,s}(\omega)$$
$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{4d}$$

$$\sum_{\{f'\in F\,:\,(f',f)\in F_c^2, j\in G_c(f',f)\}} v_{f',f,j,s}(\omega) \;\leq\; \mathbb{1}\{f\in F_{j,s}\} y_{j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_s^2, j\in G_s(f,f')\}} w_{f,f',j,s}(\omega)$$
$$\forall\, f\in F,\, j\in G,\, s\in S,\, \omega\in\Omega \tag{4e}$$

$$\sum_{j\in G_c(f_1,f_2)} \sum_{s\in S} v_{f_1,f_2,j,s}(\omega) \;\leq\; \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \qquad \forall\,(f_1,f_2)\in F_c^2,\, \omega\in\Omega \tag{4f}$$

$$\sum_{s\in S} \Bigg[\sum_{\{(i,j)\in G^2\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_s^2\}} \sum_{j\in G_s(f',f)} w_{f',f,j,s}(\omega)$$
$$- \sum_{\{f'\in F\,:\,(f,f')\in F_c^2\}} \sum_{j\in G_c(f,f')} v_{f,f',j,s}(\omega) \Bigg]$$
$$\leq \; \sum_{c\in C_f} U_c \left[M_{f,c} - \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, c\in C_{f'}\}} z_{f',f,c} - \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, c\in C_{f'}\}} z_{f,f',c} \right]$$
$$\forall\, f\in F,\, \omega\in\Omega \tag{4g}$$

$$\sum_{s\in S} \Bigg[\sum_{\{j\in G\,:\,f\in F_{j,s}\}} y_{j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f,f')\in F_s^2\}} \sum_{j\in G_s(f,f')} w_{f,f',j,s}(\omega)$$
$$- \sum_{\{f'\in F\,:\,(f',f)\in F_c^2\}} \sum_{j\in G_c(f',f)} v_{f',f,j,s}(\omega) \Bigg]$$
$$\leq \; \sum_{c\in C_f} U_c \left[M_{f,c} - \sum_{\{f'\in F\,:\,(f',f)\in F_c^2, c\in C_{f'}\}} z_{f',f,c} - \sum_{\{f'\in F\,:\,(f,f')\in F_c^2, c\in C_{f'}\}} z_{f,f',c} \right]$$
$$\forall\, f\in F,\, \omega\in\Omega \tag{4h}$$

$$\sum_{s\in S} \Bigg[\sum_{\{(i,j)\in G^2\,:\,f\in F_{ij}\}} x_{i,j,f,s}(\omega) + \sum_{\{f'\in F\,:\,(f',f)\in F_c^2\}} \sum_{j\in G_c(f',f)} v_{f',f,j,s}(\omega)$$

$$+ \sum_{\{f' \in F\,:\,(f',f)\in F_s^2\}} \sum_{j\in G_s(f',f)} w_{f',f,j,s}(\omega) \Bigg] \;\leq\; U_f \qquad \forall\, f \in F,\, \omega \in \Omega \tag{4i}$$

$$\sum_{\{f'\in F_{ij}\,:\,t_{if'}\leq t_{if}\}} x_{i,j,f',s}(\omega) \;\leq\; D_{i,j,s,t_{if}}(\omega) \qquad \forall\,(i,j)\in G_a^2,\, s\in S,\, f\in F_{ij},\, \omega\in\Omega \tag{4j}$$

$$\sum_{\left\{(f_1,f_2)\in F_c^2(a)\,:\,k_1\leq k_1(f_1,f_2)\leq k_2(f_1,f_2)\leq k_2\right\}} \sum_{c\in C_{f_1}\cap C_{f_2}} z_{f_1,f_2,c} \;\leq\; \sum_{k=k_1}^{k_2} M_k(a)$$
$$\forall\, a\in A_c,\, k_1,k_2\in T(a),\, k_1\leq k_2 \tag{4k}$$

$$\sum_{\left\{(f_1,f_2)\in F_s^2(a)\,:\,k_1\leq k_1(f_1,f_2)\leq k_2(f_1,f_2)\leq k_2\right\}} \sum_{j\in G_s(f_1,f_2)} \sum_{s\in S} w_{f_1,f_2,j,s}(\omega) \;\leq\; \sum_{k=k_1}^{k_2} W_k(a)$$
$$\forall\, a\in A_s,\, k_1,k_2\in T(a),\, k_1\leq k_2,\, \omega\in\Omega \tag{4l}$$

$$v_{f_1,f_2,j,s}(\omega) \;\in\; \mathbb{R}_+ \qquad \forall\,(f_1,f_2)\in F_c^2,\, j\in G_c(f_1,f_2),\, s\in S,\, \omega\in\Omega \tag{4m}$$

$$w_{f_1,f_2,j,s}(\omega) \;\in\; \mathbb{R}_+ \qquad \forall\,(f_1,f_2)\in F_s^2,\, j\in G_s(f_1,f_2),\, s\in S,\, \omega\in\Omega \tag{4n}$$

$$x_{i,j,f,s}(\omega) \;\in\; \mathbb{R}_+ \qquad \forall\,(i,j)\in G_a^2,\, f\in F_{ij},\, s\in S,\, \omega\in\Omega \tag{4o}$$

$$y_{j,f,s}(\omega) \;\in\; \mathbb{R}_+ \qquad \forall\, j\in G,\, s\in S,\, f\in F_{j,s},\, \omega\in\Omega \tag{4p}$$

$$z_{f_1,f_2,c} \;\in\; \mathbb{N} \qquad \forall\,(f_1,f_2)\in F_c^2,\, c\in C_{f_1}\cap C_{f_2}. \tag{4q}$$

4. Solution Method

In this section, we describe the approach used to solve problem (4). The core of the approach is a two-phase Benders decomposition algorithm. In the first phase a level bundle algorithm is used to solve the linear programming relaxation of problem (4). The linear constraints generated by the level bundle algorithm to approximate the recourse function are added to the Benders master problem at the root node of the branch-and-bound tree. In the second phase a branch-and-Benders-cut method is used to search for integer feasible first-stage solutions within a single branch-and-bound tree. Also, heuristic adaptive cuts are added to accelerate the search process.

4.1. Basic Algorithm

Note that in problem (4), the integer variables represent the first-stage decisions and the continuous variables represent the second-stage decisions. Given the first-stage decisions, the resulting problem separates into separate linear programs for different scenarios.

Let $\mathbf{Z} := \{(z_{f_1,f_2,c},\, (f_1,f_2)\in F_c^2,\, c\in C_{f_1}\cap C_{f_2}) : \text{(4b), (4k), and (4q) are satisfied}\}$ denote the set of feasible first-stage decisions. For any scenario ω, let $(4c)_\omega$, $(4d)_\omega$, $(4f)_\omega$, $(4g)_\omega$, $(4i)_\omega$, $(4j)_\omega$, and $(4l)_\omega$ denote the respective constraints (4c), (4d), (4f), (4g), (4i), (4j), and (4l) for scenario ω. For any $\bar{\mathbf{z}}\in\mathbf{Z}$ and scenario ω, let

$$\Upsilon(\bar{\mathbf{z}},\omega) \;:=\; \min_{v,w,x,y,z\geq 0} \sum_{(f_1,f_2)\in F_c^2} \sum_{j\in G_c(f_1,f_2)} \sum_{s\in S} b_{f_1,f_2,j,s} v_{f_1,f_2,j,s}(\omega) + \sum_{(f_1,f_2)\in F_s^2} \sum_{j\in G_s(f_1,f_2)} \sum_{s\in S} e_{f_1,f_2,j,s} w_{f_1,f_2,j,s}(\omega)$$

$$+ \sum_{(i,j)\in G_a^2} \sum_{f\in F_{ij}} \sum_{s\in S} h_{i,j,f,s} x_{i,j,f,s}(\omega) + \sum_{j\in G} \sum_{s\in S} \sum_{f\in F_{j,s}} q_{j,f,s} y_{j,f,s}(\omega) \tag{SP}(\bar{\mathbf{z}},\omega)$$

$$\text{s.t. } (4c)_\omega,\ (4d)_\omega,\ (4f)_\omega,\ (4g)_\omega,\ (4i)_\omega,\ (4j)_\omega,\ (4l)_\omega$$

$$z_{f_1,f_2,c} \;=\; \bar{z}_{f_1,f_2,c} \quad \forall (f_1,f_2)\in F_c^2,\ c\in C_{f_1}\cap C_{f_2} \tag{5a}$$

denote the optimal second-stage objective value. Note that for any $\bar{\mathbf{z}} \in \mathbf{Z}$ and scenario ω, the second-stage problem (SP)$(\bar{\mathbf{z}},\omega)$ is feasible and bounded. Let $\lambda(\bar{\mathbf{z}},\omega) := (\lambda_{f_1,f_2,c}(\bar{\mathbf{z}},\omega),\ (f_1,f_2)\in F_c^2,\ c\in C_{f_1}\cap C_{f_2})$ denote an extreme point optimal dual solution associated with (5a). Note that $\lambda(\bar{\mathbf{z}},\omega)\in\partial_z\Upsilon(\bar{\mathbf{z}},\omega)$, that is, $\lambda(\bar{\mathbf{z}},\omega)$ is a subgradient of $\Upsilon(\cdot,\omega)$ at $\bar{\mathbf{z}}$. Let

$$E(\mathbf{z}) \;:=\; \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c} + \sum_{\omega\in\Omega} \pi_\omega \Upsilon(\mathbf{z},\omega)$$

denote the first-stage and expected optimal second-stage objective value for given first-stage decision $\mathbf{z}\in\mathbf{Z}$. Then problem (4) can be written as

$$\min_{z\in\mathbf{Z}} \; E(\mathbf{z}) \tag{6a}$$

$$= \min \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c} + \sum_{\omega\in\Omega} \pi_\omega \alpha(\omega) \tag{MP}$$

$$\text{s.t. } (4b),\ (4k),\ (4q)$$

$$\alpha(\omega) \;\geq\; \Upsilon(\bar{\mathbf{z}},\omega) + \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} \lambda_{f_1,f_2,c}(\bar{\mathbf{z}},\omega)\,(z_{f_1,f_2,c} - \bar{z}_{f_1,f_2,c}) \quad \forall\, \bar{\mathbf{z}}\in\mathbf{Z},\ \omega\in\Omega. \tag{6b}$$

When problem (SP)$(\bar{\mathbf{z}},\omega)$ is dual degenerate, i.e., $\partial_z\Upsilon(\bar{\mathbf{z}},\omega)$ is not a singleton, then there are multiple candidates for the cuts (6b). Equation (6) holds for any choice of candidate cuts, but in Section 4.3 we discuss steps of the proposed algorithm to improve performance in the presence of dual degeneracy. The number of cuts (6b) grows exponentially in problem input, and the problem (MP) is tackled by iteratively adding cuts as needed, as described in the next section.

4.2. Two-Phase Branch-and-Benders-Cut Method

Various approaches that combine iterative cut generation and branch-and-bound can be used to solve problem (MP). For example, in one approach each major iteration i starts with cuts (6b) at a subset $\mathbf{Z}^i\subset\mathbf{Z}$ of first-stage feasible points. Then problem (MP) is solved with constraints (6b) restricted to the subset $\mathbf{Z}^i$, using a branch-and-bound algorithm. The resulting optimal solution $\mathbf{z}^i$ is added to the set $\mathbf{Z}^i$, and if the optimality gap is larger than the tolerance, the algorithm proceeds to the next major iteration. A disadvantage of this approach is that each major iteration requires solving a hard problem using a branch-and-bound algorithm, and it is typical that many iterations are needed. Similar to McDaniel and Devine (1977), Botton et al. (2013), Naoum-Sawaya

and Elhedhli (2013), and Fischetti et al. (2016), we use the following approach. A branch-and-bound algorithm starts with cuts (6b) at a set $\mathbf{Z}^0$ of points that satisfy all first-stage constraints, except that they may be fractional. The branch-and-bound algorithm proceeds by computing at each node of the branch-and-bound tree a lower bound (for a minimization problem) on the optimal objective value of problem (MP) with constraints (6b) restricted to the current subset $\mathbf{Z}^i$. If a new integer feasible solution $\mathbf{z}^i$ is found, then it is added to the current subset $\mathbf{Z}^i$. A disadvantage of this approach is that the bounds computed at nodes may be quite weak, because constraints (6b) for many of the points in $\mathbf{Z}$ are omitted at the node, and typically only a relaxation of the problem (MP) with constraints (6b) restricted to the current subset $\mathbf{Z}^i$ is solved, that is, at each node a relaxed problem with 2 "levels" of relaxation ($\mathbf{Z}^i \subset \mathbf{Z}$ and integrality requirements on the integer variables are relaxed) is solved. We use this approach because it performed better in computational experiments, mainly because it requires construction of a branch-and-bound tree only once.

Neither approach described above requires solution of the LP relaxation of problem (4). Nevertheless, we found that lower bounds obtained from the LP relaxations of problem (4) were usually remarkably strong, i.e., within a few percent of the optimal objective value of problem (4). Although the LP relaxation of problem (4) is a very large LP for realistic instances, it is much easier to solve than the original problem. Therefore, our branch-and-benders-cut method is divided into two phases. In the first phase, we use a level bundle method to solve the LP relaxation of problem (4). The cuts generated by the level bundle algorithm are included in the set of initial cuts and the corresponding points are included in the set $\mathbf{Z}^0$. The details of the level bundle method is discussed in Section 4.4. In this way the branch-and-bound algorithm in the second phase starts with a much better initial lower bound (the optimal objective value of the LP relaxation) than would have been the case without the first phase, and an initial set of cuts that give this lower bound. In addition, in the first phase a number of special cases of problem (4) with single scenarios are solved. Specifically, for each scenario $\omega \in \Omega$, the special case of problem (4) with only scenario ω is solved. In addition, the special case of problem (4) with random demand replaced by expected demand is solved. Let $E^{**}(\omega)$ denote the optimal objective value of the special case of problem (4) with only scenario ω, let $\mathbf{z}^{**}(\omega)$ denote an optimal first-stage solution for the special case of problem (4) with only scenario ω, and let $\mathbf{z}^{**}_{\text{avg}}$ denote an optimal first-stage solution for the average-scenario problem. Then $\sum_{\omega\in\Omega} \pi_\omega E^{**}(\omega)$ is an initial lower bound on the optimal objective value of problem (4), and $\min\left\{E(\mathbf{z}^{**}_{\text{avg}}), \min\{E(\mathbf{z}^{**}(\omega)) : \omega \in \Omega\}\right\}$ is an initial upper bound on the optimal objective value of problem (4).

In the second phase, the integrality requirements are imposed, and the algorithm explores a single branch-and-bound tree and adds cuts only at integer feasible solutions. Also, heuristic adaptive cuts discussed in Section 4.5 are added to accelerate the search process.

4.3. Independent Magnanti-Wong Cuts

The recourse functions $\Upsilon(\cdot,\omega)$ defined on the nonnegative reals are non-smooth, and therefore the first-stage objective function E is non-smooth, that is, there are points $\mathbf{z}$ where $\partial E(\mathbf{z})$ is not a singleton and is not continuous. Magnanti and Wong (1981) proposes selection of a nondominated subgradient (dual solution). A nondominated subgradient produces the maximum value of the resulting cut at a chosen *core point*. The proposed procedure requires the chosen core point to be in the relative interior of the convex hull of the set of feasible first-stage decisions. After solving each subproblem (SP)$(\bar{\mathbf{z}},\omega)$, the Magnanti–Wong cut generation procedure in Magnanti and Wong (1981) also requires solving an auxiliary subproblem to find the nondominated subgradient, and the auxiliary subproblem may be numerically unstable and time-consuming to solve. Also, it is time-consuming to find a core point that satisfies the requirements. As an alternative strategy to Magnanti and Wong (1981), Papadakos (2008) uses a different (approximate) core point at each step and generates nondominated cuts from an independent formulation of the Magnanti–Wong cut generation procedure.

Inspired by Papadakos (2008), we used the following approach to generate independent Magnanti–Wong cuts in addition to the usual cuts while solving the LP relaxation, which improved the solution times for problem (4). The core points are updated iteratively as described below. Let $\mathbf{z}^*_{\text{avg}}$ be an optimal first-stage solution for the LP relaxation of the average-scenario problem. Then this solution is the initial core point, $\mathbf{z}^0_{\text{core}} = \mathbf{z}^*_{\text{avg}}$. Let $\tilde{\mathbf{z}}^i$ be a first-stage solution for the LP relaxation of problem (4) produced by the level bundle master problem in iteration i of the level bundle algorithm. (The level bundle algorithm is specified in Section 4.4.) Then the core point is updated as follows:

$$\mathbf{z}^{i+1}_{\text{core}} \;=\; \frac{1}{2}\mathbf{z}^{i}_{\text{core}} + \frac{1}{2}\tilde{\mathbf{z}}^{i} \tag{7}$$

Then subproblem (SP)$(\bar{\mathbf{z}},\omega)$ is solved for $\bar{\mathbf{z}} = \mathbf{z}^{i+1}_{\text{core}}$ and each $\omega \in \Omega$, to obtain additional cuts (6b), at the point $\mathbf{z}^{i+1}_{\text{core}}$. Next, we specify the level bundle method used to solve the LP relaxation of problem (4) in the first phase of the proposed branch-and-benders-cut method.

4.4. Level Bundle Method for Solving LP Relaxation

As mentioned above, the LP relaxation of problem (4) is a very large LP for realistic instances, and it was not possible to solve the LP without using decomposition for instances with more than a fairly small number (about 10) of scenarios. Decomposition methods exploit the property that the restriction of the two-stage problem given a first-stage solution $\bar{\mathbf{z}}$ decomposes into separate (relatively small) problems for different scenarios. Various decomposition methods based on bundle methods have been proposed, including the L-shaped algorithm and decomposition methods based

on proximal bundle methods. Our computational results demonstrated a significant advantage of a decomposition algorithm based on the level bundle method over the L-shaped algorithm in terms of the computation time needed to solve the LP relaxation. The level bundle method was developed by Lemaréchal et al. (1995) for non-smooth convex optimization, and has been used to solve two-stage stochastic programs, including by Fábián and Szőke (2007).

The level bundle method requires an initial lower bound, an initial upper bound, and an initial stability center. Let γ^i_{up} and γ^i_{low} denote the upper bound and the lower bound, respectively, on the optimal objective value of the LP relaxation of problem (4), at iteration i of the level bundle algorithm. Let $\hat{\mathbf{z}}^i := \left(\hat{z}^i_{f_1,f_2,c},\ (f_1,f_2) \in F^2_c,\ c \in C_{f_1} \cap C_{f_2}\right)$ denote the stability center at iteration i of the level bundle algorithm. Let $\mathbf{z}^*(\omega)$ denote an optimal first-stage solution for the LP relaxation of the special case of problem (4) with only scenario ω, let $E^*(\omega)$ denote the optimal objective value of the LP relaxation of the special case of problem (4) with only scenario ω, and let $E(\mathbf{z}^*(\omega))$ denote the objective value of $\mathbf{z}^*(\omega)$ for the LP relaxation of problem (4). Then $\gamma^0_{\text{low}} = \sum_{\omega\in\Omega} \pi_\omega E^*(\omega)$ is an initial lower bound on the optimal objective value of the LP relaxation of problem (4). The initial stability center is $\hat{\mathbf{z}}^0 = \mathbf{z}^*_{\text{avg}}$, i.e., an optimal first-stage solution for the LP relaxation of the average-scenario problem. Also $\gamma^0_{\text{up}} = \min\left\{E(\mathbf{z}^*_{\text{avg}}), \min\{E(\mathbf{z}^*(\omega)) : \omega \in \Omega\}\right\}$ is an initial upper bound on the optimal objective value of the LP relaxation of problem (4). For each iteration i, let $\mathbf{Z}^i$ denote the set of first-stage feasible points for the LP relaxation of problem (4) at which cuts (6b) have been generated. Initially, $\mathbf{Z}^0 = \{\mathbf{z}^*_{\text{avg}}\} \cup \{\mathbf{z}^*(\omega) : \omega \in \Omega\}$.

Given input parameter $\chi \in (0,1)$, let $\gamma^i_{\text{lev}} = \chi\gamma^i_{\text{up}} + (1-\chi)\gamma^i_{\text{low}}$ denote the chosen *level* at iteration i. Let $\|\cdot\|$ denote a chosen norm. (In our computational experiments, $\|\cdot\|_2$ yielded better results than $\|\cdot\|_1$, $\|\cdot\|_3$, or $\|\cdot\|_\infty$.) Then, the level master problem at iteration i is as follows:

$$\min \sum_{(f_1,f_2)\in F^2_c} \sum_{c\in C_{f_1}\cap C_{f_2}} \frac{1}{2}\left\|z_{f_1,f_2,c} - \hat{z}^i_{f_1,f_2,c}\right\|^2 \tag{LMP}$$

$$\text{s.t. (4b), (4k)}$$

$$\sum_{(f_1,f_2)\in F^2_c} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c}$$

$$+ \sum_{\omega\in\Omega} \pi_\omega \left[\Upsilon(\bar{\mathbf{z}},\omega) + \sum_{(f_1,f_2)\in F^2_c} \sum_{c\in C_{f_1}\cap C_{f_2}} \lambda_{f_1,f_2,c}(\bar{\mathbf{z}},\omega)\left(z_{f_1,f_2,c} - \bar{z}_{f_1,f_2,c}\right)\right] \ \leq\ \gamma^i_{\text{lev}} \qquad \forall\, \bar{\mathbf{z}} \in \mathbf{Z}^i. \tag{8a}$$

$$z_{f_1,f_2,c} \ \geq\ 0 \qquad \forall\,(f_1,f_2) \in F^2_c,\ c \in C_{f_1}\cap C_{f_2} \tag{8b}$$

An important step in the level bundle method is to determine whether problem (LMP) is feasible or not. Problem (LMP-Phase1) can be solved to determine whether problem (LMP) is feasible or

not. The algorithm used to solve problem (LMP-Phase1) in level bundle iteration i can stop when a feasible solution with objective value less than γ^i_{lev} has been found.

$$\min \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c} + \sum_{\omega\in\Omega} \pi_\omega \alpha_\omega \qquad \text{(LMP-Phase1)}$$

s.t. (4b), (4k)

$$\alpha_\omega \geq \Upsilon(\bar{\mathbf{z}},\omega) + \sum_{(f_1,f_2)\in F_c^2} \sum_{c\in C_{f_1}\cap C_{f_2}} \lambda_{f_1,f_2,c}(\bar{\mathbf{z}},\omega)\left(z_{f_1,f_2,c} - \bar{z}_{f_1,f_2,c}\right) \qquad \forall\, \bar{\mathbf{z}}\in\mathbf{Z}^i,\ \omega\in\Omega \tag{9a}$$

$$z_{f_1,f_2,c} \geq 0 \qquad \forall\,(f_1,f_2)\in F_c^2,\ c\in C_{f_1}\cap C_{f_2} \tag{9b}$$

Algorithm 3 in Appendix B.1 shows the pseudo-code for the proposed level bundle method to solve the LP relaxation of problem (4).

4.5. Heuristic Adaptive Cuts

To speed up the second phase of the two-phase branch-and-Benders-cut method, we use a set of the best integer feasible first-stage solutions found so far to derive additional inequalities (not necessarily valid for problem (4)). These inequalities are updated when a new sufficiently good integer feasible solution is found.

Let $\hat{\mathbf{Z}}^j$ denote the set of first-stage feasible points for the LP relaxation of problem (4) at which cuts (6b) have been generated before the relaxation associated with the jth node of the branch-and-bound tree is solved. Initially, $\hat{\mathbf{Z}}^0 = \mathbf{Z}^{i_{\max}} \cup \left\{\mathbf{z}^{**}_{\text{avg}}\right\} \cup \{\mathbf{z}^{**}(\omega) : \omega\in\Omega\}$, where $i_{\max}$ denotes the iteration index when the level bundle method used to solve the LP relaxation of problem (4) stopped. Let $\check{\mathbf{Z}}^j$ denote the set of first-stage solutions for the LP relaxation of problem (4) that satisfy the branching constraints that apply at the jth node. Initially, $\check{\mathbf{Z}}^0 = \prod_{(f_1,f_2)\in F_c^2} \prod_{c\in C_{f_1}\cap C_{f_2}} [0,\infty)$. If, for example, the algorithm then branches by partitioning the feasible set into the subset that satisfies $z_{f'_1,f'_2,c'} = 0$ and the subset that satisfies $z_{f'_1,f'_2,c'} \geq 1$, then at the first node $\check{\mathbf{Z}}^1 = \left\{z\in\check{\mathbf{Z}}^0 : z_{f'_1,f'_2,c'} = 0\right\}$ and at the second node $\check{\mathbf{Z}}^2 = \left\{z\in\check{\mathbf{Z}}^0 : z_{f'_1,f'_2,c'} \geq 1\right\}$. Let $\tilde{\mathbf{Z}}^j$ denote the set of at most ν best integer feasible first-stage solutions found before the relaxation associated with the jth node is solved. Initially, if $\nu \geq |\Omega|+1$, then $\tilde{\mathbf{Z}}^0 = \left\{\mathbf{z}^{**}_{\text{avg}}\right\} \cup \{\mathbf{z}^{**}(\omega) : \omega\in\Omega\}$, and if $\nu < |\Omega|+1$, then $\tilde{\mathbf{Z}}^0$ contains the ν points in $\left\{\mathbf{z}^{**}_{\text{avg}}\right\} \cup \{\mathbf{z}^{**}(\omega) : \omega\in\Omega\}$ with the best objective values $E(\mathbf{z})$.

At each node j of the branch-and-bound tree, if $|\tilde{\mathbf{Z}}^j| = \nu$, then the solutions in $\tilde{\mathbf{Z}}^j$ are used to infer characteristics of good solutions. These characteristics are then imposed through the additional constraints (10)–(19) discussed below.

1. For each crossdocking flight pair $(f_1,f_2)\in F_c^2$, the number of crossdocking containers of each type $c\in C_{f_1}\cap C_{f_2}$ is forced to be between the minimum and the maximum values among all ν solutions in the set $\tilde{\mathbf{Z}}^j$.

$$z_{f_1,f_2,c} \geq \min\left\{z^h_{f_1,f_2,c} : \mathbf{z}^h\in\tilde{\mathbf{Z}}^j\right\} \qquad \forall\,(f_1,f_2)\in F_c^2,\ c\in C_{f_1}\cap C_{f_2} \tag{10}$$

$$z_{f_1,f_2,c} \leq \max\left\{z^h_{f_1,f_2,c} : \mathbf{z}^h\in\tilde{\mathbf{Z}}^j\right\} \qquad \forall\,(f_1,f_2)\in F_c^2,\ c\in C_{f_1}\cap C_{f_2} \tag{11}$$

2. For each airport $a \in A_c$ that supports crossdocking connections, the total number of crossdocking containers over all flight pairs $(f_1, f_2) \in F_c^2(a)$ and container types $c \in C_{f_1} \cap C_{f_2}$ is forced to be between the minimum and the maximum values among all ν solutions in the set $\tilde{\mathbf{Z}}^j$.

$$\sum_{(f_1,f_2)\in F_c^2(a)} \sum_{c\in C_{f_1}\cap C_{f_2}} z_{f_1,f_2,c} \;\geq\; \min\left\{ \sum_{(f_1,f_2)\in F_c^2(a)} \sum_{c\in C_{f_1}\cap C_{f_2}} z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a \in A_c \tag{12}$$

$$\sum_{(f_1,f_2)\in F_c^2(a)} \sum_{c\in C_{f_1}\cap C_{f_2}} z_{f_1,f_2,c} \;\leq\; \max\left\{ \sum_{(f_1,f_2)\in F_c^2(a)} \sum_{c\in C_{f_1}\cap C_{f_2}} z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a \in A_c \tag{13}$$

3. For each airport $a_1 \in A$ serving as the origin airport of some inbound flights that can make crossdocking connections with other outbound flights at airport $a_2 \in A_c$, let $F_{c+}^2(a_1, a_2)$ denote the set of crossdocking flight pairs $(f_1, f_2) \in F_c^2(a_2)$ with inbound flight f_1 departing from airport a_1 and arriving at airport a_2. The total crossdocking capacity provided by all flight pairs $(f_1, f_2) \in F_{c+}^2(a_1, a_2)$ is forced to be between the minimum and the maximum values among all ν solutions in the set $\tilde{\mathbf{Z}}^j$.

$$\sum_{(f_1,f_2)\in F_{c+}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\geq\; \min\left\{ \sum_{(f_1,f_2)\in F_{c+}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A,\, a_2 \in A_c \tag{14}$$

$$\sum_{(f_1,f_2)\in F_{c+}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\leq\; \max\left\{ \sum_{(f_1,f_2)\in F_{c+}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A,\, a_2 \in A_c \tag{15}$$

4. For each airport $a_2 \in A$ serving as the destination airport of some outbound flights that can make crossdocking connections with other inbound flights at airport $a_1 \in A_c$, let $F_{c-}^2(a_1, a_2)$ denote the set of crossdocking flight pairs $(f_1, f_2) \in F_c^2(a_1)$ with outbound flight f_2 departing from airport a_1 and arriving at airport a_2. The total crossdocking capacity provided by all flight pairs $(f_1, f_2) \in F_{c-}^2(a_1, a_2)$ is forced to be between the minimum and the maximum values among all ν solutions in the set $\tilde{\mathbf{Z}}^j$.

$$\sum_{(f_1,f_2)\in F_{c-}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\geq\; \min\left\{ \sum_{(f_1,f_2)\in F_{c-}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A_c,\, a_2 \in A \tag{16}$$

$$\sum_{(f_1,f_2)\in F_{c-}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\leq\; \max\left\{ \sum_{(f_1,f_2)\in F_{c-}^2(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A_c,\, a_2 \in A \tag{17}$$

5. For each airport $a_1 \in A$ serving as the origin airport of some inbound flights and each airport $a_2 \in A$ serving as the destination airport of some outbound flights, let $F^2_{c\pm}(a_1, a_2)$ denote the set of crossdocking flight pairs $(f_1, f_2) \in F^2_c$ with inbound flight f_1 departing from airport a_1 and outbound flight f_2 arriving at airport a_2. The total crossdocking capacity provided by all flight pairs $(f_1, f_2) \in F^2_{c\pm}(a_1, a_2)$ is forced to be between the minimum and the maximum values among all ν promising solutions in the set $\tilde{\mathbf{Z}}^j$.

$$\sum_{(f_1,f_2)\in F^2_{c\pm}(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\geq\; \min\left\{ \sum_{(f_1,f_2)\in F^2_{c\pm}(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A,\, a_2 \in A \tag{18}$$

$$\sum_{(f_1,f_2)\in F^2_{c\pm}(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z_{f_1,f_2,c} \;\leq\; \max\left\{ \sum_{(f_1,f_2)\in F^2_{c\pm}(a_1,a_2)} \sum_{c\in C_{f_1}\cap C_{f_2}} U_c z^h_{f_1,f_2,c} : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\} \qquad \forall\, a_1 \in A,\, a_2 \in A \tag{19}$$

Thus, at each node j of the branch-and-bound tree, if $|\tilde{\mathbf{Z}}^j| = \nu$, then the following relaxation is solved:

$$\min \sum_{(f_1,f_2)\in F^2_c} \sum_{c\in C_{f_1}\cap C_{f_2}} r_{f_1,f_2,c} z_{f_1,f_2,c} + \sum_{\omega\in\Omega} \pi_\omega \alpha_\omega \tag{RMP}$$

s.t. (4b), (4k)

$$\alpha_\omega \;\geq\; \Upsilon(\bar{\mathbf{z}}, \omega) + \sum_{(f_1,f_2)\in F^2_c} \sum_{c\in C_{f_1}\cap C_{f_2}} \lambda_{f_1,f_2,c}(\bar{\mathbf{z}}, \omega)\,(z_{f_1,f_2,c} - \bar{z}_{f_1,f_2,c}) \qquad \forall\, \bar{\mathbf{z}} \in \hat{\mathbf{Z}}^j,\, \omega \in \Omega \tag{20a}$$

$$(10)\text{—}(19) \tag{20b}$$

$$z \;\in\; \check{\mathbf{Z}}^j \tag{20c}$$

At each node j of the branch-and-bound tree, if $|\tilde{\mathbf{Z}}^j| < \nu$, then problem (RMP) without constraints (20b) is solved.

The set $\tilde{\mathbf{Z}}^j$ is updated as follows. Let $\tilde{\mathbf{z}}^j$ denote an optimal solution of problem (RMP) (with or without constraints (20b) as specified above) solved at node j of the branch-and-bound tree. If $\tilde{\mathbf{z}}^j$ is not integer feasible, then $\tilde{\mathbf{Z}}^{j+1} = \tilde{\mathbf{Z}}^j$. If $\tilde{\mathbf{z}}^j$ is integer feasible and $|\tilde{\mathbf{Z}}^j| < \nu$, then $\tilde{\mathbf{Z}}^{j+1} = \tilde{\mathbf{Z}}^j \cup \{\tilde{\mathbf{z}}^j\}$. If $\tilde{\mathbf{z}}^j$ is integer feasible and $|\tilde{\mathbf{Z}}^j| = \nu$, then let γ^j_{low} denote the overall lower bound on the optimal objective value of problem (4)) with additional inequalities (10)–(19) at the time when solution $\tilde{\mathbf{z}}^j$ is found. Let parameter $\zeta \in [0, 1]$. Let $\mathbf{z}^h_{\text{worst}} \in \arg\max\left\{E(\mathbf{z}^h) : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j\right\}$. If $E(\mathbf{z}^h_{\text{worst}}) - E(\tilde{\mathbf{z}}^j) \geq \zeta\left[E(\mathbf{z}^h_{\text{worst}}) - \gamma^j_{\text{low}}\right]$, then $\mathbf{z}^h_{\text{worst}}$ is replaced with $\tilde{\mathbf{z}}^j$ when the set $\tilde{\mathbf{Z}}^j$ is updated, that is $\tilde{\mathbf{Z}}^{j+1} = \{\tilde{\mathbf{z}}^j\} \cup \tilde{\mathbf{Z}}^j \setminus \{\mathbf{z}^h_{\text{worst}}\}$. Note that $\tilde{\mathbf{z}}^j$ satisfies constraints (20b), and therefore the feasible set of problem (RMP) for $\tilde{\mathbf{Z}}^{j+1}$ is contained in the feasible set of problem (RMP) for $\tilde{\mathbf{Z}}^j$. Thus any lower bounds computed at previous nodes remain valid. Algorithm 4 in Appendix B.3 summarizes the proposed branch-and-Benders-cut algorithm for solving problem (4).

5. Computational Results

In this section, we first present data about the express shipment system and the demand for package transportation. Then, we present results for the container crossdocking planning problem for a representative set of demand scenarios. Then, we evaluate the impact of the algorithmic features of the proposed branch-and-Benders-cut method.

5.1. Instances

The express shipment service network consists of $|G| = 282$ gateways and $|A| = 45$ airports. Based on available equipment and flight schedules, two airports denoted a_1^* and a_2^* can support both crossdocking and en-route sorting connections between flight pairs, i.e., $|A_c \cap A_s| = 2$, and one airport denoted a_3^+ can support crossdocking connections only, i.e., $|A_c \setminus A_s| = 1$, and the other 42 airports serve as "spoke" airports in the network. Table 2 in the appendix gives additional data for airports a_1^*, a_2^*, and a_3^+. Figure 8 in the appendix shows the geography of the network. There are $|S| = 2$ service levels, i.e., "next morning" (s_1) with a due time of approximately 08:00a.m. at the destination gateway, and "next day" (s_2) with a due time of approximately 01:00p.m. at the destination gateway (the due times vary somewhat by destination gateway). The revenue earned per unit delivered on-time for a specific service level is assumed to be proportional to the ground travel distance between the origin and destination gateways, and the unit revenue earned for service level s_1 is assumed to be 50% greater than that for s_2 for the same origin-destination gateway pair. There are 38618 origin-destination gateway pairs; of these 23368 origin-destination gateway pairs can provide service level s_1 and 36041 origin-destination gateway pairs can provide service level s_2.

Each gateway can always connect to its closest (by distance) airport, and may also connect to its second closest airport if its ground distance is not much greater than the distance between the gateway and its closest airport. As a result, 133 gateways are connected to their closest airports only, and 149 gateways are connected to their two closest airports. On average, the ground transportation travel times between a gateway and its closest airport and between a gateway and its second closest airport are approximately 3 hours and 5.5 hours respectively. The ground transportation costs between gateways and airports on both pickup and delivery sides are proportional to the travel distances.

The express service provider operates 172 flights between 142 pairs of airports. These flights are flown by $|B| = 3$ types of aircraft (denoted b_1, b_2 and b_3). The flight network satisfies flow balance, that is, for each airport and each aircraft type, the number of flights that arrive at the airport is equal to the number of flights that depart from the airport.

There are $|C| = 5$ types of containers (denoted $c_1, \ldots, c_5$). The maximum total weight of packages in kilograms that each container type can carry are $U_{c_1} = 1510$, $U_{c_2} = 1690$, $U_{c_3} = 480$, $U_{c_4} = 1340$, and $U_{c_5} = 760$, respectively.

The express service provider also signs contracts with passenger airlines for package transportation capacity on passenger flights. This involves 3331 passenger flights for 942 origin-destination airport pairs. The average and standard deviation of available capacity on a passenger flight are 651 and 243 kilograms, respectively. The average and standard deviation of flight duration of a passenger flight are 161 and 65 minutes, respectively. The express service provider does not have sufficient control over the loading of freight on passenger flights to use passenger flights for crossdocking connections. In contrast, en-route sorting connections are allowed between two flights of any type, i.e., self-operated flights or passenger flights, if time permits. A crossdocking connection takes about 1 hour and an en-route sorting connection takes about 2 hours. The number of self-operated flight pairs that can be connected through crossdocking and through en-route sorting are 3020 and 2455, respectively. On average, a self-operated flight can connect to 35 and to 30 self-operated flights through crossdocking and through en-route sorting, respectively. There are 5949 additional en-route sorting connections that can be made between two passenger flights or between a passenger flight and a self-operated flight. The number of first-stage variables is equal to 4381.

The crossdocking cost $r_{f_1,f_2,c}$ per container of type c and sorting cost per kilogram $e_{f_1,f_2,j,s}$ are assumed to be the same for different flight pairs (f_1, f_2), different destination gateways j, and different service levels s, and if flight pairs $(f_1, f_2) \in F_c^2(a)$ and $(f_1', f_2') \in F_s^2(a)$ connect at an airport $a \in A_c \cap A_s$, then their values satisfy $r_{f_1,f_2,c} = e_{f_1',f_2',j,s} U_c/2$. In other words, if the weight of packages carried by a container of type $c \in C_{f_1} \cap C_{f_2}$ is less than half of the container capacity U_c, then it costs less to make an en-route sorting connection than to do crossdocking between f_1 and f_2 (if the sorting capacity at the connecting airport and the time between the flights allow).

Demand data for each origin-destination gateway pair and each service level were recorded in time intervals of 10 minutes each according to the time interval during which the packages become available for transportation after being sorted at the origin gateway. Thus, each day was partitioned into 144 time intervals of 10 minutes each, and each demand category is characterized by an origin gateway, a destination gateway, a service level, and one of the 144 time intervals during which the packages become available for transportation. In each scenario, the amount of demand in each category is represented by the weight of packages in that category.

5.2. Results

In this section we present some computational results, and additional computational results are presented in Appendix E.

5.2.1. Computational Performance Results Results for solving the LP relaxation of problem (4) with the level bundle method and the L-shaped method are shown in Appendix E.1. Figure 3 shows the upper bound γ^j_{up}, the lower bound γ^j_{low}, and the number of first-stage variables with values fixed by constraints (20b), versus elapsed computation time in hours after the LP relaxation of problem (4) has been solved. The proposed method found a feasible solution with better objective value than the initial incumbent solution (obtained from the average-scenario problem) fairly quickly. That resulted in updates to the set $\tilde{\mathbf{Z}}^j$ of good solutions, which in turn resulted in an increase in the number of first-stage variables with values fixed by constraints (20b). Later, around 19 hours of computing, further improvements in the incumbent solution resulted in further updates to the set $\tilde{\mathbf{Z}}^j$, and another noticeable increase in the number of first-stage variables with values fixed by constraints (20b). This was followed by more noticeable increases in the lower bounds, resulting in the gap between the upper bound and the lower bound becoming smaller than 0.1%. The gap between the objective value of the final solution and the optimal objective value of the LP relaxation of problem (4) was 1.9%, thereby showing that although the additional constraints (20b) might result in some loss of optimality, the final solution was within 1.9% of optimal. The gap between the objective value of the final solution and the optimal objective value with no crossdocking containers (all first-stage variable values equal to 0) was 10.8%, showing the benefit of using crossdocking connections between eligible flight pairs.

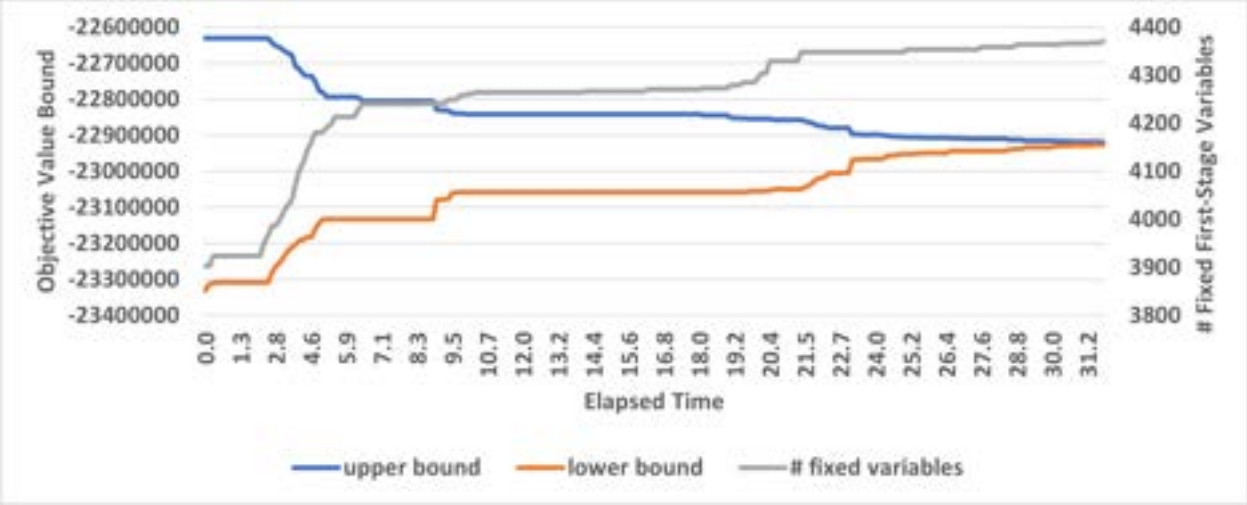


Figure 3 **Computational performance results of proposed branch-and-Benders-cut algorithm with additional adaptive cuts for solving problem** (4)**.**

To evaluate the effect of the heuristic adaptive cuts (20b), a variant of Algorithm 4 without the heuristic adaptive cuts (20b) was tested. Figure 4 shows the objective values of integer feasible solutions found by this variant, the upper bound (the dashed line that remains constant at

the initial upper bound $\gamma^0_{\text{up}} = \min\left\{E(\mathbf{z}^*_{\text{avg}}), \min\{E(\mathbf{z}^*(\omega)) : \omega \in \Omega\}\right\}$), and the lower bound γ^j_{low}, versus elapsed computation time in hours after the LP relaxation of problem (4) has been solved. This algorithm failed to improve the initial upper bound, that is, although the heuristic adaptive cuts (20b) reduces the feasible set, it enables the algorithm to find better feasible solutions faster. The initial lower bound was improved by only an amount (1365) too small to notice in the figure. Similar performance was observed when we used this algorithm to solve problem (4) for other sets of scenarios, i.e., the initial upper bound could not be improved by the algorithm, and the improvement of the lower bound was very small.

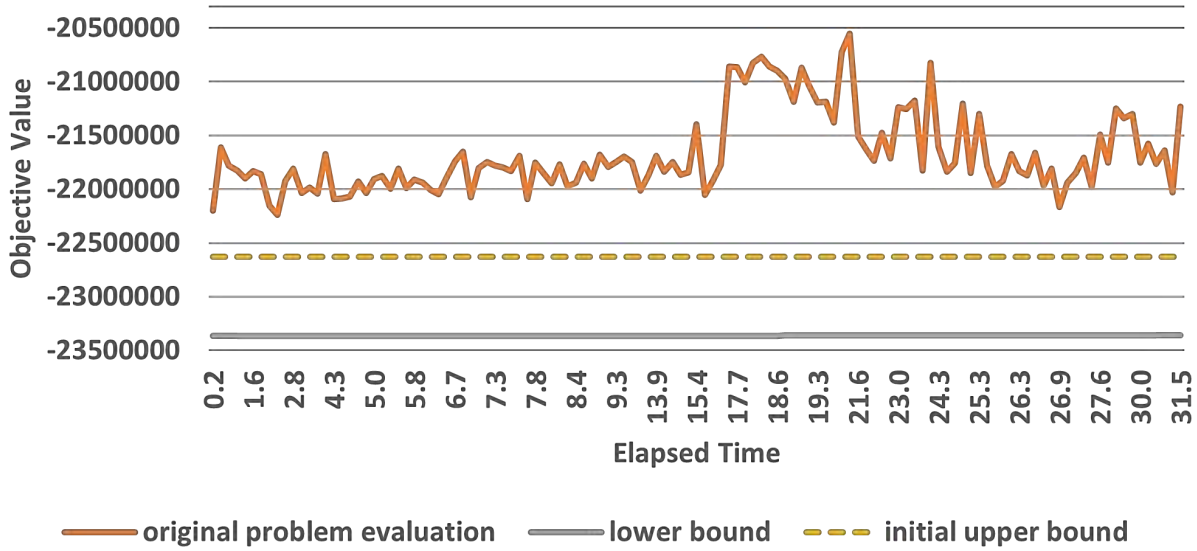
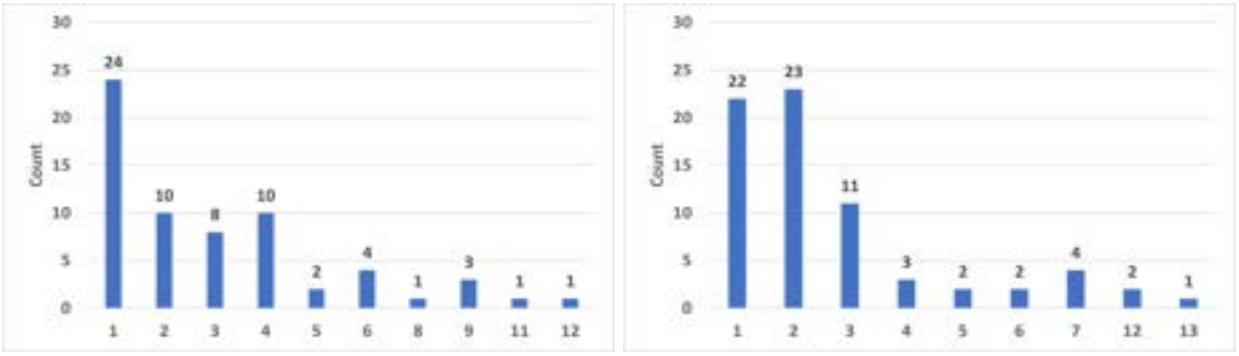


Figure 4 **Computational performance results of branch-and-Benders-cut algorithm without additional adaptive cuts for solving problem** (4)**.**

5.2.2. Optimal Solution Characteristics Next we report some key characteristics of the best feasible solution found by Algorithm 4.

In the first-stage solution produced by Algorithm 4, 242 containers make crossdocking connections between 200 pairs of flights and between 177 pairs of airports. Figure 5a shows the distribution of the number of outbound flights connected by crossdocking to each inbound flight, and Figure 5b shows the distribution of the number of inbound flights connected by crossdocking to each outbound flight. Most inbound flights are connected by crossdocking to 1–4 outbound flights, and most outbound flights are connected by crossdocking to 1–3 inbound flights, but one inbound flight is connected by crossdocking to 12 outbound flights, and one outbound flight is connected by crossdocking to 13 inbound flights.



(a) The distribution of the number of outbound flights connected by crossdocking to each inbound flight.

(b) The distribution of the number of inbound flights connected by crossdocking to each outbound flight.

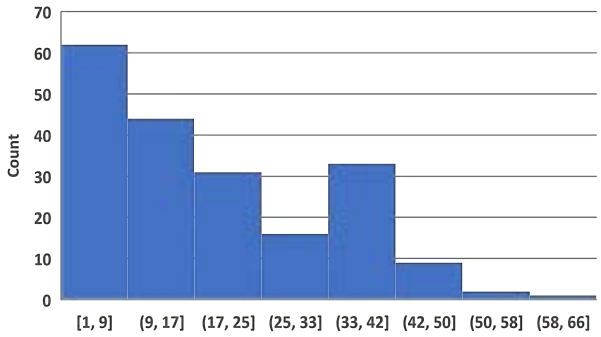
Figure 5 The distribution of the number of flights connected by crossdocking.

Table 1 shows, for each scenario ω, the total service level s_1 shipment flows W_{s_1}, the total service level s_2 shipment flows W_{s_2}, the total service level s_1 demand that is not served W'_{s_1}, the total service level s_2 demand that is not served W'_{s_2}, the proportion p_{s_1} of service level s_1 demand served, the proportion p_{s_2} of service level s_2 demand served, the proportion p'_c of demand that is served through crossdocking connections, the proportion p'_s of demand that is served through en-route sorting connections, and the proportion p'_d of demand that is served through direct flights. The proportion of demand served through crossdocking connections is smaller for scenarios with larger total demand.

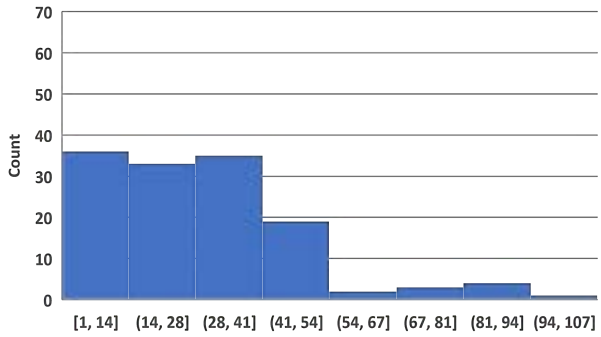
Table 1 Summary of shipment flows.

ω	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
W_{s_1}	420731	478389	487827	540839	536675	601929	620665	697108	667699	666843
W_{s_2}	965480	1090864	1100620	1229878	1222943	1347552	1352238	1490946	1459696	1438630
W'_{s_1}	25229	34020	33673	64297	44637	59775	71517	93099	91479	89082
W'_{s_2}	121642	180339	185751	238485	215699	295088	331618	445955	404502	404887
p_{s_1}	94.3%	93.4%	93.5%	89.4%	92.3%	91.0%	89.7%	88.2%	88.0%	88.2%
p_{s_2}	88.8%	85.8%	85.6%	83.8%	85.0%	82.0%	80.3%	77.0%	78.3%	78.0%
p'_c	22.2%	21.5%	21.3%	20.7%	20.4%	19.4%	19.4%	17.8%	18.3%	18.3%
p'_s	28.8%	30.0%	29.9%	31.1%	30.6%	32.3%	30.8%	34.2%	32.8%	33.8%
p'_d	48.9%	48.4%	48.8%	48.2%	48.9%	48.3%	49.8%	48.0%	49.0%	47.9%
ω	ω_{11}	ω_{12}	ω_{13}	ω_{14}	ω_{15}	ω_{16}	ω_{17}	ω_{18}	ω_{19}	ω_{20}
W_{s_1}	661628	723699	714778	714502	706857	710803	763268	772876	754010	883595
W_{s_2}	1428369	1542825	1573031	1576419	1552253	1574347	1630365	1645103	1648469	1727091
W'_{s_1}	82445	121328	120600	114672	125357	117773	143762	148167	123830	198920
W'_{s_2}	386653	523156	496282	466767	488589	461349	574238	615640	532838	924687
p_{s_1}	88.9%	85.6%	85.6%	86.2%	84.9%	85.8%	84.2%	83.9%	85.9%	81.6%
p_{s_2}	78.7%	74.7%	76.0%	77.2%	76.1%	77.3%	74.0%	72.8%	75.6%	65.1%
p'_c	18.5%	17.2%	17.3%	17.3%	17.5%	17.3%	16.6%	16.6%	16.5%	15.4%
p'_s	32.9%	34.3%	35.7%	35.1%	34.1%	35.1%	35.5%	35.0%	35.1%	31.9%
p'_d	48.7%	48.5%	47.0%	47.7%	48.4%	47.6%	47.8%	48.4%	48.4%	52.6%

There are 3829 pairs of flights that serve some demand (in at least one demand scenario) through en-route sorting connections between 1606 pairs of airports. Figure 6a shows the distribution of the number of outbound flights connected by en-route sorting to each inbound flight, and Figure 6b shows the distribution of the number of inbound flights connected by en-route sorting to each outbound flight.



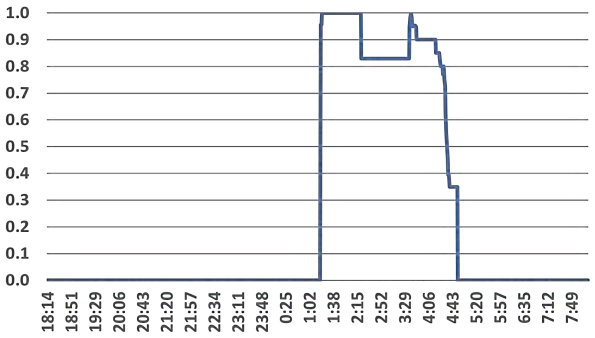
(a) The distribution of the number of outbound flights connected by en-route sorting to each inbound flight.



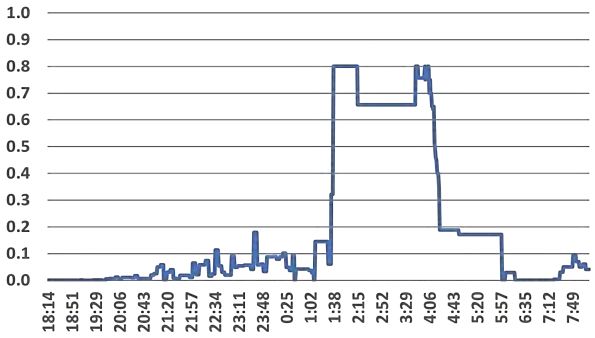
(b) The distribution of the number of inbound flights connected by en-route sorting to each outbound flight.

Figure 6 **The distribution of the number of flights connected by en-route sorting.**

Figure 7 shows the average en-route sorting capacity utilization over time at airports a_1^* and a_2^*. Most en-route sorting takes place between 01:00a.m. and 4:30a.m. when most self-operated flights arrive at and depart from the two airports.



(a) En-route sorting capacity utilization at airport a_1^*.



(b) En-route sorting capacity utilization at airport a_2^*.

Figure 7 **En-route sorting capacity utilization over time at airports that support en-route sorting.**

6. Conclusion

The use of container crossdocking in air transportation is of great importance for reducing processing time and handling cost. A challenge with the use of container crossdocking is that the number of containers crossdocking between successive pairs of flights has to be planned before the amount of freight available to use the crossdocking containers becomes known. In addition, there is substantial variation in origin-destination freight flows from day to day. We proposed a two-stage optimization model that can be used to plan the air transportation operations of an express shipment service, including container crossdocking and en-route sorting, in the presence of randomly varying demand. Conventional methods were not able to solve the optimization problem, and therefore we proposed and tested a number of algorithmic strategies, including solving a number of easier optimization problems in the first phase to start the branch-and-Benders-cut procedure in the second phase with better bounds and with better cuts, as well as adaptively using characteristics of good solutions found so far to add cuts that improve bounds and make it easier to find other good solutions. Computational experiments demonstrate that these algorithmic strategies greatly improve the quality of the solutions found with the allocated computational budget. We demonstrated that the proposed approach can produce good plans for the air transportation operations of an express shipment service, even in the presence of substantial uncertainty about demand.

Acknowledgments

We thank SF Express for introducing us to several fascinating problems associated with express shipment transportation, including the problem addressed in this paper.

References

M. Aboytes-Ojeda, K. K. Castillo-Villar, and M. S. Roni. A decomposition approach based on meta-heuristics and exact methods for solving a two-stage stochastic biofuel hub-and-spoke network problem. *Journal of Cleaner Production*, 247:119176, 2020.

D. Agustina, C. Lee, and R. Piplani. A review: Mathematical modles for cross docking planning. *International Journal of Engineering Business Management*, 2(2):47–54, 2010.

A. P. Armacost, C. Barnhart, and K. A. Ware. Composite variable formulations for express shipment service network design. *Transportation science*, 36(1):1–20, 2002.

G. Bayraksan. Solving stochastic programs. *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

N. Boland, M. Fischetti, M. Monaci, and M. Savelsbergh. Proximity benders: a decomposition heuristic for stochastic programs. *Journal of Heuristics*, 22(2):181–198, 2016.

Q. Botton, B. Fortz, L. Gouveia, and M. Poss. Benders decomposition for the hop-constrained survivable network design problem. *INFORMS journal on computing*, 25(1):13–26, 2013.

C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83(1):451–464, 1998.

China State Post Bureau. China express development index report 2021. `http://www.spb.gov.cn/gjyzj/c100278/202205/4f2917df7dfc4ba28c31949456eb390e.shtml` (06-04-2022), 2022.

C. I. Fábián and Z. Szőke. Solving two-stage stochastic programming problems with level decomposition. *Computational Management Science*, 4(4):313–353, 2007.

M. Fischetti, I. Ljubić, and M. Sinnl. Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research*, 253(3):557–569, 2016.

H. Fleuren, C. Goossens, M. Hendriks, M.-C. Lombard, I. Meuffels, and J. Poppelaars. Supply chain–wide optimization at tnt express. *Interfaces*, 43(1):5–20, 2013.

M. J. Kuby and R. G. Gray. The hub network design problem with stopovers and feeders: The case of federal express. *Transportation Research Part A: Policy and Practice*, 27(1):1–12, 1993.

A.-L. Ladier and G. Alpan. Cross-docking operations: Current research versus industry practice. *Omega*, 62:145–162, 2016.

G. Laporte and F. V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations research letters*, 13(3):133–142, 1993.

C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical programming*, 69(1):111–147, 1995.

J. Linderoth and S. Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2):207–250, 2003.

I. Louwerse, J. Mijnarends, I. Meuffels, D. Huisman, and H. Fleuren. Scheduling movements in the network of an express service provider. *Flexible Services and Manufacturing Journal*, 26(4):565–584, 2014.

T. L. Magnanti and R. T. Wong. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, 29(3):464–484, 1981.

D. McDaniel and M. Devine. A modified benders' partitioning algorithm for mixed integer programming. *Management Science*, 24(3):312–319, 1977.

J. Naoum-Sawaya and S. Elhedhli. An interior-point benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research*, 210(1):33–55, 2013.

N. Papadakos. Practical enhancements to the magnanti–wong method. *Operations Research Letters*, 36(4): 444–449, 2008.

J. M. Q. Pérez, J.-S. Tancrez, and J.-C. Lange. Express shipment service network design with complex routes. *Computers & Operations Research*, 114:104810, 2020.

R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei. The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei. Accelerating the benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization*, 28(1):875–903, 2018.

A. J. Rubiales, P. A. Lotito, and L. A. Parente. Stabilization of the generalized benders decomposition applied to short-term hydrothermal coordination problem. *IEEE Latin America Transactions*, 11(5): 1212–1224, 2013.

A. Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical programming*, 35(3):309–333, 1986.

T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.

B. Torbali and G. Alpan. A literature review on robust and real-time models for cross-docking. *International Journal of Production Research*, pages 1–30, 2022.

W. van Ackooij, A. Frangioni, and W. de Oliveira. Inexact stabilized benders' decomposition approaches to chance-constrained problems with finite support, 2015.

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

H. Wu, M. Savelsbergh, and Y. Huang. Planning the city operations of a parcel express company. *Omega*, 107:102539, 2022.

B. Yıldız and M. Savelsbergh. Optimizing package express operations in china. *European Journal of Operational Research*, 300(1):320–335, 2022.

Appendix A: Proofs

A.1. Proof of Proposition 1

Proof. Consider any $a \geq 0$, any $b \in B(a)$, and any $x \in X(a,b)$. Then, for every $U \subset T$, $U \neq \varnothing$ it holds that

$$\sum_{j \in U} b_j \;=\; \sum_{j \in U} \sum_{i \in \delta_j^-} x_{ij} \;\leq\; \sum_{i \in \bigcup\limits_{j \in U} \delta_j^-} a_i$$

and thus $b \in H(a)$. Hence $B(a) \subset H(a)$.

Next we show that $H(a) \subset B(a)$. Consider any $b \notin B(a)$, $b \geq 0$. Then

$$\begin{aligned} 0 \;<\; \min\; & \sum_{j \in T} e_j && && (21)\\ \text{s.t.}\; & \sum_{j \in \delta_i^+} x_{ij} \;\leq\; a_i && \forall\, i \in S\\ & \sum_{i \in \delta_j^-} x_{ij} + e_j \;=\; b_j && \forall\, j \in T\\ & x_{ij} \;\geq\; 0 && \forall\, (i,j) \in A\\ & e_j \;\geq\; 0 && \forall\, j \in T \end{aligned}$$

Consider any optimal solution (x^*, e^*) of (21). Then execute Algorithm 1 with input (x^*, e^*). Note that a

Algorithm 1 Find violated subset constraint.

Require: Optimal solution (x^*, e^*) of (21)

1: Choose any $j^0 \in T$ such that $e^*_{j^0} > 0$, and set $U^0 = \{j^0\}$
2: Set $\ell = 0$
3: **while** $\sum_{j \in U^\ell} b_j \leq \sum_{i \in \bigcup_{j \in U^\ell} \delta_j^-} a_i$ **do**
4: Find any $j^{\ell+1} \in T \setminus U^{\ell^*}$ such that $\sum_{i \in \left(\bigcup_{j \in U^\ell} \delta_j^-\right) \cap \delta^-_{j^{\ell+1}}} x^*_{i,j^{\ell+1}} > 0$, and set $U^{\ell+1} = U^\ell \cup \{j^{\ell+1}\}$
5: Increment $\ell \leftarrow \ell + 1$
6: **end while**
7: Output U^ℓ

j^0 as specified in Step 1 of Algorithm 1 exists. Next we show by contradiction that, in each iteration ℓ, if $\sum_{j \in U^\ell} b_j \leq \sum_{i \in \bigcup_{j \in U^\ell} \delta_j^-} a_i$, then a $j^{\ell+1}$ as specified in Step 4 of Algorithm 1 exists. Suppose there is an iteration ℓ^* such that $\sum_{i \in \left(\bigcup_{j \in U^{\ell^*}} \delta_j^-\right) \cap \delta^-_{j'}} x^*_{i,j'} = 0$ for all $j' \in T \setminus U^{\ell^*}$.

Case 1: It holds that $\sum_{j \in \delta_i^+} x^*_{i,j} = a_i$ for all $i \in \bigcup_{j \in U^{\ell^*}} \delta_j^-$.

Then

$$\sum_{i \in \bigcup\limits_{j \in U^{\ell^*}} \delta_j^-} a_i \;=\; \sum_{i \in \bigcup\limits_{j \in U^{\ell^*}} \delta_j^-} \sum_{j \in \delta_i^+} x^*_{i,j}$$

$$
\begin{aligned}
&= \sum_{i \in \bigcup\limits_{j \in U^{\ell^*}} \delta_j^-} \left(\sum_{j \in U^{\ell^*} \bigcap \delta_i^+} x^*_{i,j} + \sum_{j \in (T \setminus U^{\ell^*}) \bigcap \delta_i^+} x^*_{i,j} \right) \\
&= \sum_{j \in U^{\ell^*}} \sum_{i \in \delta_j^-} x^*_{i,j} + \sum_{j' \in T \setminus U^{\ell^*}} \sum_{i \in \left(\bigcup\limits_{j \in U^{\ell^*}} \delta_j^- \right) \bigcap \delta_{j'}^-} x^*_{i,j'} \\
&= \sum_{j \in U^{\ell^*}} \sum_{i \in \delta_j^-} x^*_{i,j} \\
&= \sum_{j \in U^{\ell^*}} \left(b_j - e^*_j \right) \;<\; \sum_{j \in U^{\ell^*}} b_j
\end{aligned}
$$

where the inequality follows from $e^*_{j^0} > 0$. The inequality contradicts condition $\sum_{j \in U^\ell} b_j \le \sum_{i \in \bigcup_{j \in U^\ell} \delta_j^-} a_i$ in Step 3 of Algorithm 1, and thus Case 1 cannot occur.

Case 2: There is an $i^* \in \bigcup_{j \in U^{\ell^*}} \delta_j^-$ such that $\sum_{j \in \delta_{i^*}^+} x^*_{i^*,j} < a_{i^*}$.

Choose any $j^* \in U^{\ell^*} \cap \delta_{i^*}^+$. Step 4 of Algorithm 1 implies that for each $\ell \in \{1, \dots, \ell_*\}$, there is an $\ell' \in \{0, \dots, \ell - 1\}$ and an $i \in \delta_{j^{\ell'}}^- \cap \delta_{j^\ell}^-$ such that $x^*_{i,j^\ell} > 0$. Therefore, there is a sequence $0 = \ell_0 < \ell_1 < \cdots < \ell_k \le \ell^*$ and a sequence $i^0, \dots, i^{k-1}$ such that $j^{\ell_k} = j^*$, and for each $l \in \{0, \dots, k-1\}$ it holds that $i^l \in \delta_{j^{\ell_l}}^-$ and $x^*_{i^l, j^{\ell_{l+1}}} > 0$. Let $\varepsilon := \min\{e^*_{j^0},\, a_{i^*} - \sum_{j \in \delta_{i^*}^+} x^*_{i^*,j},\, \min\{x^*_{i^0,j^{\ell_1}}, \dots, x^*_{i^{k-1},j^{\ell_k}}\}\} > 0$. Then increase $x^*_{i^*,j^*} = x^*_{i^*,j^{\ell_k}}$ by ε (note that $\sum_{j \in \delta_{i^*}^+} x^*_{i^*,j} + \varepsilon \le a_{i^*}$), and decrease $x^*_{i^{k-1},j^{\ell_k}}$ by ε (note that $x^*_{i^{k-1},j^{\ell_k}} - \varepsilon \ge 0$). Similarly, for each $l \in \{1, \dots, k-1\}$, increase $x^*_{i^l,j^{\ell_l}}$ by ε, and decrease $x^*_{i^{l-1},j^{\ell_l}}$ by ε. Last, increase $x^*_{i^0,j^{\ell_0}}$ by ε, and decrease $e^*_{j^0}$ by ε. The resulting solution is feasible for (21), and has strictly better objective value than (x^*, e^*). The contradiction establishes that Case 2 cannot occur.

Algorithm 1 terminates after at most T iterations with a set $U^\ell \subset T$ such that $\sum_{j \in U^\ell} b_j > \sum_{i \in \bigcup_{j \in U^\ell} \delta_j^-} a_i$. Thus $b \notin H(a)$, and hence $H(a) \subset B(a)$. Therefore $H(a) = B(a)$, and hence a subset of the inequalities $b \ge 0$; $\sum_{j \in U} b_j \le \sum_{i \in \bigcup_{j \in U} \delta_j^-} a_i$, $U \subset T$, $U \ne \varnothing$ are the facet-defining inequalities of $B(a)$. □

A.2. Proof of Proposition 2

Proof. Note that $H(a) \subset P(a)$.

Next we show that $P(a) \subset H(a)$. Consider any $a \ge 0$ and $b \notin H(a) = B(a)$, $b \ge 0$. Then (21) holds. Consider any optimal solution (x^*, e^*) of (21), and execute Algorithm 2 with input (x^*, e^*). Note that a j^0 as specified in Step 1 of Algorithm 2 exists. Next we show by contradiction that, in each iteration ℓ, if $\sum_{\{j \in T\,:\, k_1^\ell \le k_1(j) \le k_2(j) \le k_2^\ell\}} b_j \le \sum_{i=k_1^\ell}^{k_2^\ell} a_i$, then a $j^{\ell+1}$ as specified in Step 4 of Algorithm 2 exists. Suppose there is an iteration ℓ^* such that no $j^{\ell^*+1} \in T$ exists such that $\left(k_1(j^{\ell^*+1}) < k_1^{\ell^*} \text{ or } k_2(j^{\ell^*+1}) > k_2^{\ell^*}\right)$ and $\sum_{i=\max\{k_1^{\ell^*},\, k_1(j^{\ell^*+1})\}}^{\min\{k_2^{\ell^*},\, k_2(j^{\ell^*+1})\}} x^*_{i,j^{\ell^*+1}} > 0$.

Case 1: It holds that $\sum_{j \in \delta_i^+} x^*_{i,j} = a_i$ for all $i \in \left\{k_1^{\ell^*}, \dots, k_2^{\ell^*}\right\}$.

Then

$$
\begin{aligned}
\sum_{i=k_1^{\ell^*}}^{k_2^{\ell^*}} a_i &= \sum_{i=k_1^{\ell^*}}^{k_2^{\ell^*}} \sum_{j \in \delta_i^+} x^*_{i,j} \\
&= \sum_{i=k_1^{\ell^*}}^{k_2^{\ell^*}} \left(\sum_{\{j \in \delta_i^+\,:\, k_1^{\ell^*} \le k_1(j) \le k_2(j) \le k_2^{\ell^*}\}} x^*_{i,j} + \sum_{\{j \in \delta_i^+\,:\, k_1(j) < k_1^{\ell^*} \text{ or } k_2(j) > k_2^{\ell^*}\}} x^*_{i,j} \right)
\end{aligned}
$$

Algorithm 2 Find violated interval constraint.

Require: Optimal solution (x^*, e^*) of (21)

1: Choose any $j^0 \in T$ such that $e^*_{j^0} > 0$, and set $k_1^0 = k_1(j^0)$ and $k_2^0 = k_2(j^0)$

2: Set $\ell = 0$

3: **while** $\sum_{\{j \in T\,:\, k_1^\ell \le k_1(j) \le k_2(j) \le k_2^\ell\}} b_j \le \sum_{i=k_1^\ell}^{k_2^\ell} a_i$ **do**

4: Find any $j^{\ell+1} \in T$ such that $\big(k_1(j^{\ell+1}) < k_1^\ell$ or $k_2(j^{\ell+1}) > k_2^\ell\big)$ and $\sum_{i=\max\{k_1^\ell,\, k_1(j^{\ell+1})\}}^{\min\{k_2^\ell,\, k_2(j^{\ell+1})\}} x^*_{i,j^{\ell+1}} > 0$, and set $k_1^{\ell+1} = \min\{k_1^\ell,\, k_1(j^{\ell+1})\}$ and $k_2^{\ell+1} = \max\{k_2^\ell,\, k_2(j^{\ell+1})\}$

5: Increment $\ell \leftarrow \ell + 1$

6: **end while**

7: Output k_1^ℓ, k_2^ℓ

$$
\begin{aligned}
&= \sum_{\{j \in T\,:\, k_1^{\ell^*} \le k_1(j) \le k_2(j) \le k_2^{\ell^*}\}} \sum_{i=k_1(j)}^{k_2(j)} x^*_{i,j} + \sum_{\{j \in \delta_i^+\,:\, k_1(j) < k_1^{\ell^*} \text{ or } k_2(j) > k_2^{\ell^*}\}} \sum_{i=\max\{k_1^{\ell^*},\, k_1(j)\}}^{\min\{k_2^{\ell^*},\, k_2(j)\}} x^*_{i,j} \\
&= \sum_{\{j \in T\,:\, k_1^{\ell^*} \le k_1(j) \le k_2(j) \le k_2^{\ell^*}\}} \sum_{i=k_1(j)}^{k_2(j)} x^*_{i,j} \\
&= \sum_{\{j \in T\,:\, k_1^{\ell^*} \le k_1(j) \le k_2(j) \le k_2^{\ell^*}\}} \big(b_j - e^*_j\big) \;<\; \sum_{\{j \in T\,:\, k_1^{\ell^*} \le k_1(j) \le k_2(j) \le k_2^{\ell^*}\}} b_j
\end{aligned}
$$

where the inequality follows from $e^*_{j^0} > 0$. The inequality contradicts condition $\sum_{\{j \in T\,:\, k_1^\ell \le k_1(j) \le k_2(j) \le k_2^\ell\}} b_j \le \sum_{i=k_1^\ell}^{k_2^\ell} a_i$ in Step 3 of Algorithm 2, and thus Case 1 cannot occur.

Case 2: There is an $i^* \in \big\{k_1^{\ell^*}, \ldots, k_2^{\ell^*}\big\}$ such that $\sum_{j \in \delta_{i^*}^+} x^*_{i^*,j} < a_{i^*}$.

Case 2.1: $i^* \in \{k_1^0, \ldots, k_2^0\}$.

Note that $i^* \in \delta_{j^0}^-$. Let $\varepsilon := \min\{e^*_{j^0},\, a_{i^*} - \sum_{j \in \delta_{i^*}^+} x^*_{i^*,j}\} > 0$. Then increase $x^*_{i^*,j^0}$ by ε (note that $\sum_{j \in \delta_{i^*}^+} x^*_{i^*,j} + \varepsilon \le a_{i^*}$), and decrease $e^*_{j^0}$ by ε (note that $e^*_{j^0} - \varepsilon \ge 0$). The resulting solution is feasible for (21), and has strictly better objective value than (x^*, e^*). The contradiction establishes that Case 2.1 cannot occur.

Case 2.2: $i^* \notin \{k_1^0, \ldots, k_2^0\}$.

By Step 4 of Algorithm 2, $k_1^{\ell+1} \le k_1^\ell$ and $k_2^{\ell+1} \ge k_2^\ell$ for all ℓ. Also, $i^1 := i^* \in \big\{k_1^{\ell^*}, \ldots, k_2^{\ell^*}\big\}$. Thus, there is an $\ell_1 \in \{1, \ldots, \ell^*\}$ such that $i^1 \in \big\{k_1^{\ell_1}, \ldots, k_2^{\ell_1}\big\}$ and $i^1 \notin \big\{k_1^{\ell_1 - 1}, \ldots, k_2^{\ell_1 - 1}\big\}$. Then by Step 4 of Algorithm 2, j^{ℓ_1} satisfies $\big(k_1(j^{\ell_1}) < k_1^{\ell_1 - 1}$ or $k_2(j^{\ell_1}) > k_2^{\ell_1 - 1}\big)$ and $\sum_{i=\max\{k_1^{\ell_1-1},\, k_1(j^{\ell_1})\}}^{\min\{k_2^{\ell_1-1},\, k_2(j^{\ell_1})\}} x^*_{i,j^{\ell_1}} > 0$. Since $i^1 \in \big\{k_1^{\ell_1}, \ldots, k_2^{\ell_1}\big\}$ and $i^1 \notin \big\{k_1^{\ell_1 - 1}, \ldots, k_2^{\ell_1 - 1}\big\}$, it follows that $i^1 \in \delta_{j^{\ell_1}}^-$. Also, since $\sum_{i=\max\{k_1^{\ell_1-1},\, k_1(j^{\ell_1})\}}^{\min\{k_2^{\ell_1-1},\, k_2(j^{\ell_1})\}} x^*_{i,j^{\ell_1}} > 0$, there is an $i^2 \in \big\{\max\{k_1^{\ell_1-1}, k_1(j^{\ell_1})\}, \ldots, \min\{k_2^{\ell_1-1}, k_2(j^{\ell_1})\}\big\}$ such that $x^*_{i^2,j^{\ell_1}} > 0$. Since $i^1 \notin \big\{k_1^{\ell_1-1}, \ldots, k_2^{\ell_1-1}\big\}$, it follows that $i^1 \ne i^2$. Either $i^2 \in \{k_1^0, \ldots, k_2^0\}$ in which case $\ell_2 = 0$, or there is an $\ell_2 \in \{1, \ldots, \ell_1 - 1\}$ such that $i^2 \in \big\{k_1^{\ell_2}, \ldots, k_2^{\ell_2}\big\}$ and $i^2 \notin \big\{k_1^{\ell_2 - 1}, \ldots, k_2^{\ell_2 - 1}\big\}$. Proceeding in this way, there is a sequence $i^* = i^1, \ldots, i^k$ and a sequence $\ell^* \ge \ell_1 > \cdots > \ell_k = 0$ such that for each $l \in \{1, \ldots, k\}$ it holds that $i^l \in \big\{k_1^{\ell_l}, \ldots, k_2^{\ell_l}\big\}$ and for each $l \in \{1, \ldots, k-1\}$ it holds that $x^*_{i^{l+1},j^{\ell_l}} > 0$. Let $\varepsilon := \min\{e^*_{j^0},\, a_{i^*} - \sum_{j \in \delta_{i^*}^+} x^*_{i^*,j},\, \min\{x^*_{i^2,j^{\ell_1}}, \ldots, x^*_{i^k,j^{\ell_{k-1}}}\}\} > 0$. Then increase $x^*_{i^*,j^{\ell_1}} = x^*_{i^1,j^{\ell_1}}$ by ε (recall that $i^1 \in \delta_{j^{\ell_1}}^-$, and note that $\sum_{j \in \delta_{i^*}^+} x^*_{i^*,j} + \varepsilon \le a_{i^*}$), and decrease $x^*_{i^2,j^{\ell_1}}$ by ε (note that $x^*_{i^2,j^{\ell_1}} - \varepsilon \ge 0$). Similarly, for each $l \in \{2, \ldots, k-1\}$, increase $x^*_{i^l,j^{\ell_l}}$ by ε, and decrease

$x^*_{i^{l+1},j^{\ell_l}}$ by ε. Last, increase $x^*_{i^k,j^{\ell_k}} = x^*_{i^k,j^0}$ by ε, and decrease $e^*_{j^0}$ by ε. The resulting solution is feasible for (21), and has strictly better objective value than (x^*, e^*). The contradiction establishes that Case 2.2 cannot occur.

Algorithm 2 terminates after at most T iterations with k_1^ℓ, k_2^ℓ such that $\sum_{\{j \in T\,:\, k_1^\ell \le k_1(j) \le k_2(j) \le k_2^\ell\}} b_j > \sum_{i=k_1^\ell}^{k_2^\ell} a_i$. Thus $b \notin P(a)$, and hence $P(a) \subset H(a)$. Therefore $H(a) = P(a)$. □

Appendix B: Additional Algorithmic Details

B.1. Level Bundle Method for Solving the LP Relaxation of Problem (4)

Algorithm 3 Level Bundle Algorithm for Solving the LP Relaxation of Problem (4).

Input: Stopping tolerance $\varepsilon > 0$, control parameters $\chi \in (0,1)$ and $\theta \in (0,1)$;
Solve LP relaxations of single-scenario problems, obtain $\mathbf{z}^*(\omega)$ and $E^*(\omega)$ for each $\omega \in \Omega$;
Solve (SP)$(\bar{\mathbf{z}}, \omega)$ for $\bar{\mathbf{z}} = \mathbf{z}^*(\omega')$, obtain $\Upsilon(\mathbf{z}^*(\omega'), \omega)$ and $\lambda(\mathbf{z}^*(\omega'), \omega)$, for each $\omega, \omega' \in \Omega$;
Solve LP relaxation of average-scenario problem, obtain $\mathbf{z}^*_{\text{avg}}$;
Solve (SP)$(\bar{\mathbf{z}}, \omega)$ for $\bar{\mathbf{z}} = \mathbf{z}^*_{\text{avg}}$, obtain $\Upsilon(\mathbf{z}^*_{\text{avg}}, \omega)$ and $\lambda(\mathbf{z}^*_{\text{avg}}, \omega)$, for each $\omega \in \Omega$;
$\mathbf{Z}^0 = \{\mathbf{z}^*_{\text{avg}}\} \cup \{\mathbf{z}^*(\omega) : \omega \in \Omega\}$;
$\hat{\mathbf{z}}^0 = \mathbf{z}^*_{\text{avg}}$, $\mathbf{z}^0_{\text{core}} = \mathbf{z}^*_{\text{avg}}$;
$\gamma^0_{\text{low}} = \sum_{\omega \in \Omega} \pi_\omega E^*(\omega)$, $\gamma^0_{\text{up}} = \min\left\{E(\mathbf{z}^*_{\text{avg}}), \min\{E(\mathbf{z}^*(\omega)) : \omega \in \Omega\}\right\}$; $\gamma^0_{\text{lev}} = \chi\gamma^0_{\text{up}} + (1-\chi)\gamma^0_{\text{low}}$;
for $i = 0, 1, \ldots$ **do**
  if $(\gamma^i_{\text{up}} - \gamma^i_{\text{low}})/|\gamma^i_{\text{up}}| \le \varepsilon$ **then**
    return γ^i_{low}, γ^i_{up}, and $\hat{\mathbf{z}}^i$;
  end if
  Run an algorithm to optimize (LMP-Phase1) until either its lower bound $\underline{\gamma}^i$ is greater than γ^i_{lev} or its upper bound $\overline{\gamma}^i$ is less than γ^i_{lev};
  if the lower bound $\underline{\gamma}^i$ on the optimal objective value of (LMP-Phase1) is greater than γ^i_{lev}, that is, (LMP) is infeasible **then**
    $\hat{\mathbf{z}}^{i+1} = \hat{\mathbf{z}}^i$, $\mathbf{z}^{i+1}_{\text{core}} = \mathbf{z}^i_{\text{core}}$, $\mathbf{Z}^{i+1} = \mathbf{Z}^i$;
    $\gamma^{i+1}_{\text{low}} = \underline{\gamma}^i$, $\gamma^{i+1}_{\text{up}} = \gamma^i_{\text{up}}$, $\gamma^{i+1}_{\text{lev}} = \chi\gamma^{i+1}_{\text{up}} + (1-\chi)\gamma^{i+1}_{\text{low}}$;
  else
    Solve (LMP) and get an optimal solution $\tilde{\mathbf{z}}^i$ for (LMP);
    Solve (SP)$(\bar{\mathbf{z}}, \omega)$ for $\bar{\mathbf{z}} = \tilde{\mathbf{z}}^i$, obtain $\Upsilon(\tilde{\mathbf{z}}^i, \omega)$ and $\lambda(\tilde{\mathbf{z}}^i, \omega)$ for each $\omega \in \Omega$, compute $E(\tilde{\mathbf{z}}^i)$;
    if $E(\tilde{\mathbf{z}}^i) \le (1-\theta)\gamma^i_{\text{up}} + \theta\gamma^i_{\text{low}}$ **then**
      $\hat{\mathbf{z}}^{i+1} = \tilde{\mathbf{z}}^i$, $\gamma^{i+1}_{\text{lev}} = \chi E(\tilde{\mathbf{z}}^i) + (1-\chi)\gamma^i_{\text{low}}$;
    else
      $\hat{\mathbf{z}}^{i+1} = \hat{\mathbf{z}}^i$, $\gamma^{i+1}_{\text{lev}} = \gamma^i_{\text{lev}}$;
    end if
    $\gamma^{i+1}_{\text{low}} = \gamma^i_{\text{low}}$, $\gamma^{i+1}_{\text{up}} = \min\{\gamma^i_{\text{up}}, E(\tilde{\mathbf{z}}^i)\}$, $\mathbf{z}^{i+1}_{\text{core}} = \frac{1}{2}\mathbf{z}^i_{\text{core}} + \frac{1}{2}\tilde{\mathbf{z}}^i$, $\mathbf{Z}^{i+1} = \mathbf{Z}^i \cup \{\tilde{\mathbf{z}}^i, \mathbf{z}^{i+1}_{\text{core}}\}$;
    Solve (SP)$(\bar{\mathbf{z}}, \omega)$ for $\bar{\mathbf{z}} = \mathbf{z}^{i+1}_{\text{core}}$, obtain $\Upsilon(\mathbf{z}^{i+1}_{\text{core}}, \omega)$ and $\lambda(\mathbf{z}^{i+1}_{\text{core}}, \omega)$, for each $\omega \in \Omega$;
  end if
end for

B.2. Solving Sub-Problems in Parallel

In the proposed Algorithm 3 to solve the LP relaxation of problem (4), as well as in the Branch-and-Benders-Cut method described in Section 4.2, subproblem (SP)$(\bar{\mathbf{z}},\omega)$ is solved multiple times for some $\bar{\mathbf{z}}$ and for each $\omega\in\Omega$. For any given $\bar{\mathbf{z}}$, subproblems (SP)$(\bar{\mathbf{z}},\omega)$ for different $\omega\in\Omega$ are very similar — they differ only in the right side values in constraints $(4\mathrm{j})_\omega$. After subproblem (SP)$(\bar{\mathbf{z}},\omega)$ has been solved for some $\bar{\mathbf{z}}$ and some $\omega\in\Omega$, using re-optimization techniques subproblem (SP)$(\bar{\mathbf{z}},\omega)$ for the same $\bar{\mathbf{z}}$ and another $\omega'\in\Omega$ can be solved in just a few more iterations. Numerical experiments showed that sorting scenarios $\omega\in\Omega$ in non-decreasing order of total demand and then solving subproblems (SP)$(\bar{\mathbf{z}},\omega)$ sequentially requires much shorter solution times than solving the subproblems independently. On the other hand, when multiple processors are available, then subproblems (SP)$(\bar{\mathbf{z}},\omega)$ can be solved in parallel. Also, in experiments the simplex algorithm outperformed the barrier or interior-point algorithms for solving the subproblems. Since the simplex algorithm in commercial solvers requires only one processor, one can use the simplex algorithm to solve different subproblems in parallel, where the maximum number of subproblems that can be solved simultaneously is equal to the number of processors. In our experiments, the number of subproblems to be solved at each step was not larger than the number of available processors. Solving the subproblems in parallel was found to reduce computation time by 86% relative to solving the subproblems independently, and by 69% relative to solving the subproblems in sequence with the above mentioned re-optimization approach.

B.3. Branch-and-Benders-Cut Method with Additional Adaptive Cuts for Solving Problem (4)

Algorithm 4 Branch-and-Benders-Cut Algorithm with Additional Adaptive Cuts for Solving Problem (4).

input: Stopping tolerance $\varepsilon>0$, control parameters ν and ζ;
Solve the LP relaxation of problem (4) with Algorithm 3, and get LP optimal objective value LP*;
Solve single-scenario instance of problem (4) for each $\omega\in\Omega$, obtain $\mathbf{z}^{**}(\omega)$, $E^{**}(\omega)$, and $E(\mathbf{z}^{**}(\omega))$;
Solve the average-scenario instance of problem (4), obtain $\mathbf{z}^{**}_{\text{avg}}$ and $E(\mathbf{z}^{**}_{\text{avg}})$;
Set initial incumbent solution $\hat{\mathbf{z}}^0\in\arg\min\big\{E(\mathbf{z}^{**}_{\text{avg}}),\min\{E(\mathbf{z}^{**}(\omega)):\omega\in\Omega\}\big\}$, initial upper bound $\gamma^0_{\text{up}}=\min\big\{E(\mathbf{z}^{**}_{\text{avg}}),\min\{E(\mathbf{z}^{**}(\omega)):\omega\in\Omega\}\big\}$, and initial lower bound $\gamma^0_{\text{low}}=\max\big\{\text{LP}^*,\sum_{\omega\in\Omega}\pi_\omega E^{**}(\omega)\big\}$;
Set $\hat{\mathbf{Z}}^0=\mathbf{Z}^{i_{\max}}\cup\big\{\mathbf{z}^{**}_{\text{avg}}\big\}\cup\{\mathbf{z}^{**}(\omega):\omega\in\Omega\}$;
if $\nu\geq\big|\big\{\mathbf{z}^{**}_{\text{avg}}\big\}\cup\{\mathbf{z}^{**}(\omega):\omega\in\Omega\}\big|$ **then**
    $\tilde{\mathbf{Z}}^0=\big\{\mathbf{z}^{**}_{\text{avg}}\big\}\cup\{\mathbf{z}^{**}(\omega):\omega\in\Omega\}$;
else
    $\tilde{\mathbf{Z}}^0$ contains the ν points in $\big\{\mathbf{z}^{**}_{\text{avg}}\big\}\cup\{\mathbf{z}^{**}(\omega):\omega\in\Omega\}$ with the best objective values $E(\mathbf{z})$;
end if
Set $\check{\mathbf{Z}}^0=\prod_{(f_1,f_2)\in F^2_c}\prod_{c\in C_{f_1}\cap C_{f_2}}[0,\infty)$;
for $j=0,1,\ldots$ **do**
    if $\gamma^j_{\text{up}}\leq\gamma^j_{\text{low}}+\varepsilon$ **then**
        return γ^j_{low}, γ^j_{up}, and $\hat{\mathbf{z}}^j$;
    end if

if $|\tilde{\mathbf{Z}}^j| = \nu$ **then**
  Solve problem (RMP), get $\tilde{\mathbf{z}}^j$, and update $\gamma_{\text{low}}^{j+1}$;
else
  Solve problem (RMP) without constraints (20b), get $\tilde{\mathbf{z}}^j$, and update $\gamma_{\text{low}}^{j+1}$;
end if
if $\tilde{\mathbf{z}}^j$ is integer feasible **then**
  Solve (SP)$(\bar{\mathbf{z}}, \omega)$ for $\bar{\mathbf{z}} = \tilde{\mathbf{z}}^j$, obtain $\Upsilon(\tilde{\mathbf{z}}^j, \omega)$ and $\lambda(\tilde{\mathbf{z}}^j, \omega)$ for each $\omega \in \Omega$, obtain $E(\tilde{\mathbf{z}}^j)$;
  $\hat{\mathbf{Z}}^{j+1} = \hat{\mathbf{Z}}^j \cup \{\tilde{\mathbf{z}}^j\}$;
  if $E(\tilde{\mathbf{z}}^j) < \gamma_{\text{up}}^j$ **then**
    $\gamma_{\text{up}}^{j+1} = E(\tilde{\mathbf{z}}^j)$, $\hat{\mathbf{z}}^{j+1} = \tilde{\mathbf{z}}^j$;
  else
    $\gamma_{\text{up}}^{j+1} = \gamma_{\text{up}}^j$, $\hat{\mathbf{z}}^{j+1} = \hat{\mathbf{z}}^j$;
  end if
  if $|\tilde{\mathbf{Z}}^j| = \nu$ **then**
    $\mathbf{z}_{\text{worst}}^h \in \arg\max \left\{ E(\mathbf{z}^h) : \mathbf{z}^h \in \tilde{\mathbf{Z}}^j \right\}$;
    if $E(\mathbf{z}_{\text{worst}}^h) - E(\tilde{\mathbf{z}}^j) \geq \zeta \left[E(\mathbf{z}_{\text{worst}}^h) - \gamma_{\text{low}}^j \right]$ **then**
      $\tilde{\mathbf{Z}}^{j+1} = \{\tilde{\mathbf{z}}^j\} \cup \tilde{\mathbf{Z}}^j \setminus \{\mathbf{z}_{\text{worst}}^h\}$;
    else
      $\tilde{\mathbf{Z}}^{j+1} = \tilde{\mathbf{Z}}^j$;
    end if
  else
    $\tilde{\mathbf{Z}}^{j+1} = \tilde{\mathbf{Z}}^j \cup \{\tilde{\mathbf{z}}^j\}$;
  end if
else
  $\gamma_{\text{up}}^{j+1} = \gamma_{\text{up}}^j$, $\hat{\mathbf{z}}^{j+1} = \hat{\mathbf{z}}^j$, $\tilde{\mathbf{Z}}^{j+1} = \tilde{\mathbf{Z}}^j$;
  Choose next branching;
end if
Choose next node $j+1$, and set $\check{\mathbf{Z}}^{j+1}$ accordingly;
end for

Appendix C: Additional Instance Data

The container crossdocking capacity and package sorting capacity (if applicable) per unit time for each airport $a \in A_c$ is assumed to be constant. Table 2 reports for each airport $a \in A_c$ the container crossdocking capacity $M(a)$ in containers per 24 hours, the sorting capacity $W(a)$ (if applicable) in kilograms per minute, the average ($\bar{\tau}^+(a)$) and standard deviation ($\tilde{\tau}^+(a)$) of air travel times (in minutes) from airport a to all the other airports in the network, and the average ($\bar{\tau}^-(a)$) and standard deviation ($\tilde{\tau}^-(a)$) of air travel times (in minutes) from all the other airports in the network to airport a.

Table 2 Major airport data.

Airport $a \in A_c$	$M(a)$	$W(a)$	$\bar{\tau}^+(a)$	$\tilde{\tau}^+(a)$	$\bar{\tau}^-(a)$	$\tilde{\tau}^-(a)$
a_1^*	500	2400	112	36	115	36
a_2^*	240	600	147	59	138	55
a_3^+	50	—	146	55	158	60

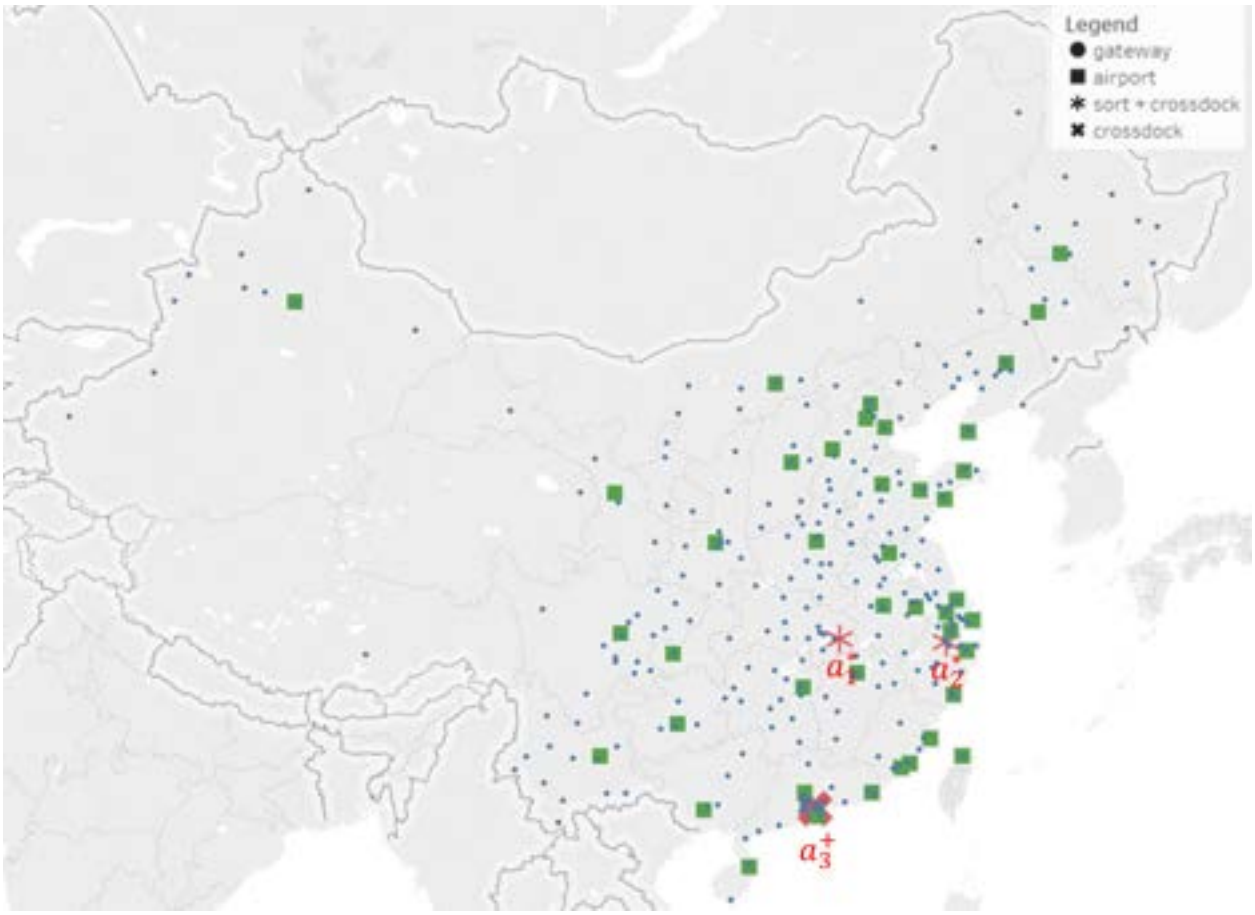


Figure 8 Express service provider gateway and airport network.

Table 3 shows, for each aircraft type $b \in B$, the number $|F_b|$ of flights flown by the aircraft type b, the resulting flight capacity U_f in kilograms, the resulting number $M_{f,c}$ of containers of type $c \in C$ that the flight can accommodate, and the average ($\bar{\tau}_b$) and the standard deviation ($\tilde{\tau}_b$) of flight duration in minutes among flights $f \in F_b$ in the network.

Table 3 Aircraft type data.

Aircraft type $b \in B$	$\lvert F_b \rvert$	U_f	M_{f,c_1}	M_{f,c_2}	M_{f,c_3}	M_{f,c_4}	M_{f,c_5}	$\bar{\tau}_b$	$\tilde{\tau}_b$
b_1	90	14000	0	8	1	0	0	98	46
b_2	68	25000	0	14	0	1	0	106	36
b_3	14	48000	22	2	0	0	15	99	28

Figure 9 shows the geographic distribution of flights operated by the express service provider with each type of aircraft. Figure 10 shows the number F_b of flights operated by the express service provider with each aircraft type b (denoted by $|F_{b_1}|/|F_{b_2}|/|F_{b_3}|$) that arrive at (and depart from) each airport in the network. Note that the number of flights that arrive at airports $a \in A_c$ (highlighted with red color) is much higher than at the other airports.

(a) Flights operated by the express service provider with aircraft type b_1.

(b) Flights operated by the express service provider with aircraft type b_2.

(c) Flights operated by the express service provider with aircraft type b_3.

Figure 9 The flights operated by the express service provider with each type of aircraft.

Figure 11 shows arrival times and departure times of flights operated by the express service provider at each major airport $a \in A_c$. Flights arrive before 03:00a.m. and depart after 03:00a.m.

Figure 12 shows the demands for a single origin-destination gateway pair $(i, j) \in G^2$ and for service levels s_1 and s_2 that become available in each of the 144 time intervals of 10 minutes each, for 2 scenarios. These demands and the cutoff times t for flights are used to compute the cumulative demands $D_{i,j,s,t}(\omega)$ as input for an instance of problem (4).

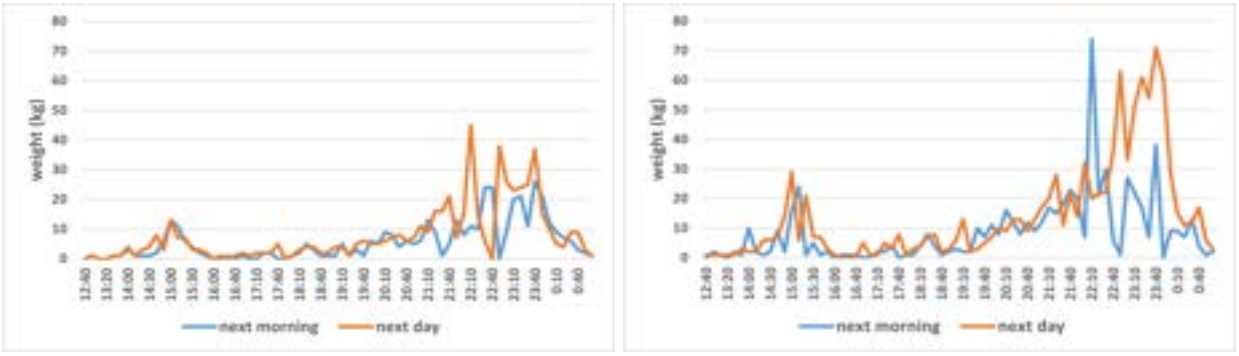
Figure 13a shows the Lorenz curve of the cumulative proportion of total demand (total over both service levels and time intervals, and averaged over different scenarios) against the cumulative proportion of origin-destination gateway pairs (sorted in non-increasing order of total demand). It shows that the top 1% and top

Figure 10 **Number of flights operated by the express service provider at each airport.**

Figure 11 **Arrival times and departure times of flights operated by the express service provider at each major airport.**

10% of origin-destination gateway pairs contribute approximately 31% and 78% of total demand, respectively. Figure 13b illustrates the relative demand (total over both service levels and time intervals, and averaged over different scenarios) associated with each gateway (which serves either as demand origin or destination), where gateways with greater total demand have larger circle sizes in the figure.

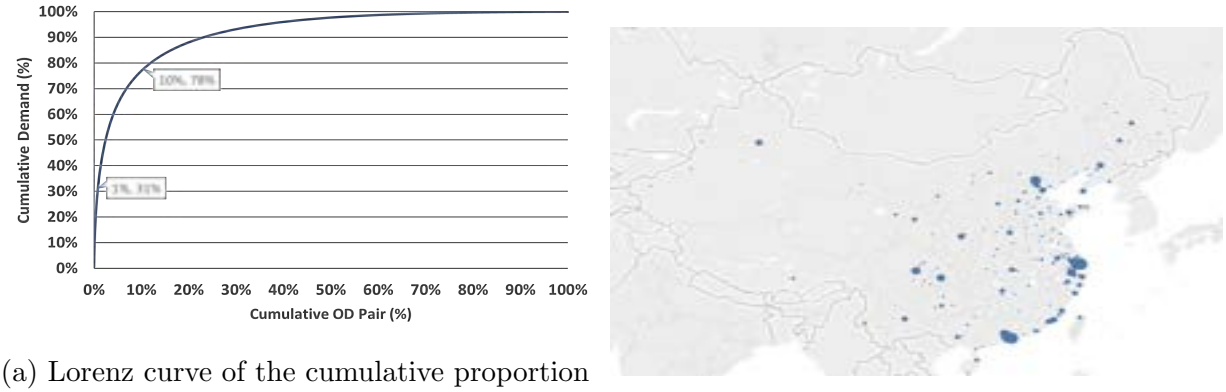
Figure 14 shows the total demand (total over all origin-destination pairs and time intervals) for both service levels and for each scenario. Next morning demand and next day demand contribute approximately 29% and 71% of total demand in each scenario, respectively. Note that although some of the scenarios (such as ω_2 and ω_3) have similar total demand, the amount of demand for specific origin-destination gateway pairs and specific time intervals are quite different between these scenarios. The total demand in scenario ω_{20}



(a) Demands for a single origin-destination pair and for the two service levels that become available in each 10 minute interval in scenario ω_2.

(b) Demands for a single origin-destination pair and for the two service levels that become available in each 10 minute interval in scenario ω_{19}.

Figure 12 **Demands for a single origin-destination pair and for the two service levels that become available in each** 10 **minute interval in two scenarios.**



(a) Lorenz curve of the cumulative proportion of total demand versus the cumulative proportion of origin-destination pairs.

(b) The relative demand associated with each gateway.

Figure 13 **Spatial distribution of demand.**

is about 250% of the total demand in scenario ω_1, and thus there is quite large demand variation among different scenarios.

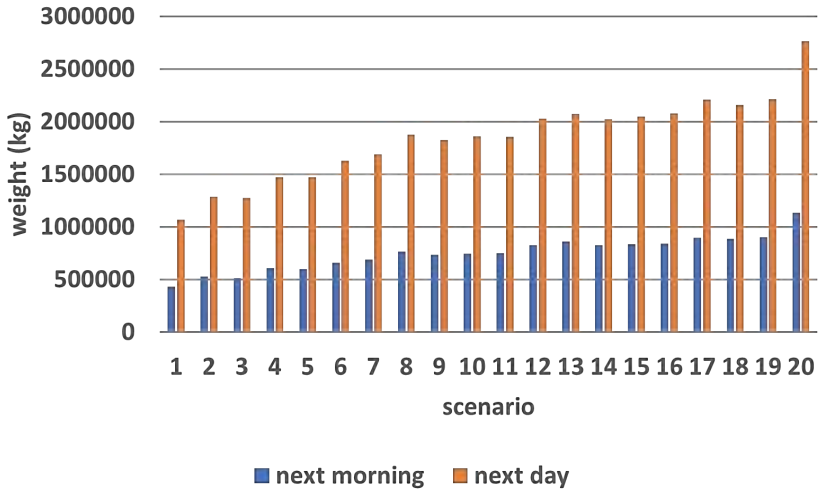


Figure 14 **Total demand for each service level in each scenario.**

Appendix D: Computational Parameters

The branch-and-Benders-cut algorithm, including the level bundle algorithm to solve the LP relaxation, was coded in Python and used Gurobi 9.0.3 to solve linear and mixed-integer programs. All experiments were performed in a dedicated server system with 40 cores of x86-64 processing (4 Intel Xeon ES-2650 v3 @ 2.3GHz processors) and 256GB RAM running Red Hat Enterprise Linux Server 7.4.

We conducted experiments to calibrate the values of the algorithm parameters seeking to find a good trade-off between the time allocated to different phases of the proposed branch-and-Benders-cut algorithm and the quality of the results. We observed that the values $\varepsilon = 0.001$, $\chi = 0.35$, $\theta = 0.1$, $\nu = 7$, and $\zeta = 0.04$ generally resulted in a good trade-off between algorithm speed and solution quality. Also, the maximum search time of the second phase of the proposed method was chosen such that the entire program would finish within 48 hours. Python's multiprocessing package was used for process-based parallel computing for solving the single-scenario and average-scenario problems and for solving the second-stage subproblems, where Gurobi parameters "Method" and "Threads" were both set to 1 so that a single thread was used by the dual simplex method to solve the second-stage subproblem in each parallel process. During the second phase of the branch-and-Benders-cut algorithm, the addition of cuts at new integer feasible solutions and the update of the right side values of heuristic adaptive cuts (10)–(19) were achieved by setting Gurobi parameter "LazyConstraints" to 1 and adding lazy constraints through a callback function within Gurobi.

Appendix E: Additional Results

E.1. Evaluation of Solution Strategies

The aim of this part of the computational experiments was to evaluate the effect of each of the proposed algorithmic strategies.

E.1.1. Effect of Solving LP Relaxation Before Branch-and-Benders-Cut In this section we explore the effect of the two-stage approach proposed in Algorithm 4 that solves the LP relaxation of problem (4) before starting the branch-and-Benders-cut search, versus a pure branch-and-Benders-cut search without first solving the LP relaxation. Recall that Figure 3 shows the upper bound γ^j_{up}, the lower bound γ^j_{low}, and the number of first-stage variables with values fixed by constraints (20b), versus elapsed computation time in hours, obtained with the two-stage approach proposed in Algorithm 4. Similarly, Figure 15 shows the upper bound γ^j_{up}, the lower bound γ^j_{low}, and the number of first-stage variables with values fixed by constraints (20b), versus elapsed computation time in hours, obtained with a pure branch-and-Benders-cut search without first solving the LP relaxation. For easy comparison, a dashed line shows the best upper bound found by the two-stage approach. Note that the initial lower bound without first solving the LP relaxation is given by $\gamma^0_{\text{low}} = \sum_{\omega\in\Omega} \pi_\omega E^{**}(\omega)$. The optimal objective value LP* of the LP relaxation satisfied LP$^* > \sum_{\omega\in\Omega} \pi_\omega E^{**}(\omega)$, and thus solving the LP relaxation provides a stronger initial lower bound γ^0_{low}. Also, although the lower bound γ^j_{low} resulting from solving problem (RMP) includes additional constraints (20b), it alone was not sufficient for the lower bound γ^j_{low} to improve the lower bound LP* (without the additional constraints). Thus, solving the LP relaxation of problem (4) before starting the branch-and-Benders-cut search substantially improves the lower bound.

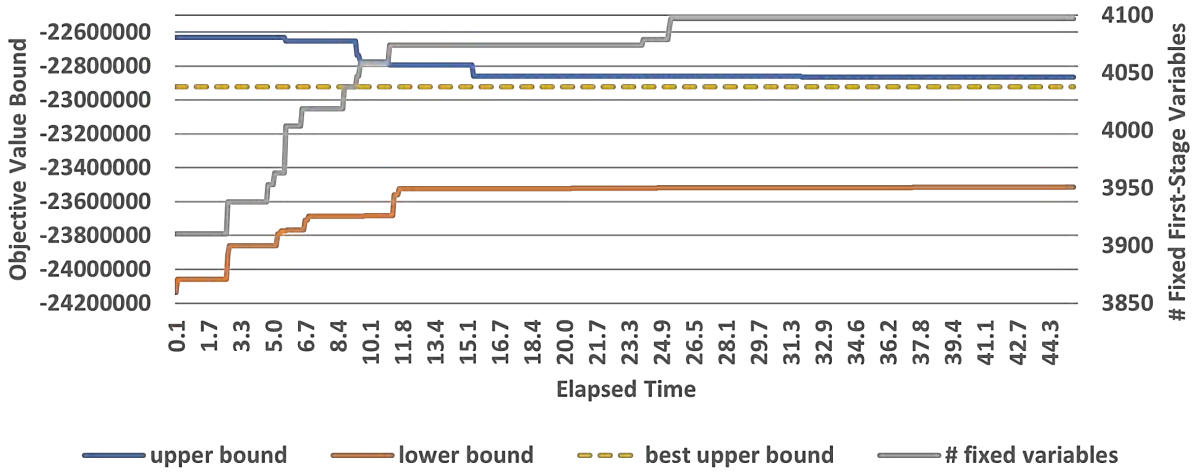


Figure 15 Computational performance results of branch-and-Benders-cut algorithm for solving problem (4) without first solving the LP Relaxation of problem (4).

The effect of solving the LP relaxation of problem (4) before starting the branch-and-Benders-cut search was evaluated for 10 different randomly generated sets $\Omega_1, \ldots, \Omega_{10}$ of scenarios, each set Ω_n containing

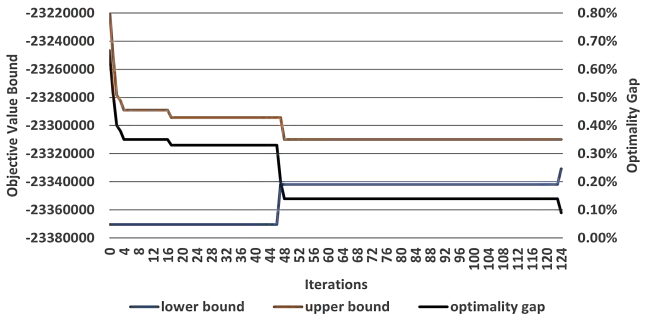
$|\Omega_n| = 20$ scenarios. Table 4 shows the gap $\text{gap}^2_{\text{avg}} = \left[E(\mathbf{z}^{**}_{\text{avg}}) - \gamma^0_{\text{low}}\right] / |\gamma^0_{\text{low}}|)$ between the objective value $E(\mathbf{z}^{**}_{\text{avg}})$ for problem (4) of the first-stage solution of the average-scenario problem and the LP relaxation of problem (4), the gap gap^3 between the objective value of the best integer feasible solution found in 48 hours of computing by the branch-and-Benders-cut search without solving the LP relaxation first and the lower bound given by problem (RMP), and the gap gap^4 between the objective value of the best integer feasible solution found in 48 hours of computing by the branch-and-Benders-cut search without solving the LP relaxation first and the LP relaxation of problem (4), for each set of scenarios. (Note that the same time budget of 48 hours was allocated to the branch-and-Benders-cut search without solving the LP relaxation first as to the branch-and-Benders-cut search including the time to solve the LP relaxation first.)

Table 4 Effect of solving the LP relaxation of problem (4) before starting the branch-and-Benders-cut search on 10 sets of scenarios.

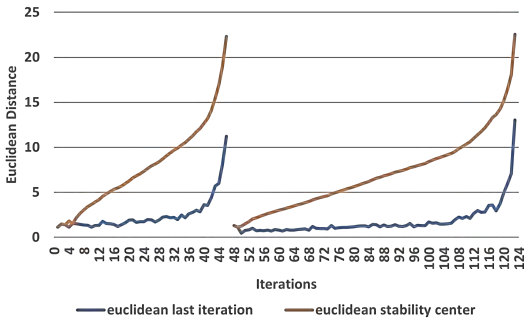
Ω		Ω_1	Ω_2	Ω_3	Ω_4	Ω_5	Ω_6	Ω_7	Ω_8	Ω_9	Ω_{10}
$\text{gap}^2_{\text{avg}}$		3.1%	3.5%	4.1%	3.0%	3.7%	3.9%	3.6%	3.7%	4.2%	3.2%
two-stage	gap^3	0.2%	0.4%	0.8%	0.2%	0.9%	0.3%	0.7%	0.1%	0.4%	0.7%
	gap^4	1.9%	2.6%	2.0%	1.7%	1.8%	2.3%	2.2%	2.0%	2.1%	2.5%
one-stage	gap^3	2.8%	3.5%	4.3%	3.2%	2.5%	3.3%	2.9%	4.1%	3.7%	3.4%
	gap^4	2.0%	2.8%	2.5%	2.6%	1.9%	2.9%	2.4%	2.8%	3.0%	2.8%

E.1.2. Performance of Level Bundle Method versus L-Shaped Method Figure 16a shows the lower bound, the upper bound, and the optimality gap versus iteration index for the level bundle method. Figure 16b shows (1) the Euclidean distance between the level master problem optimal solution $\tilde{\mathbf{z}}^i$ in each iteration and the stability center $\hat{\mathbf{z}}^i$ in the same iteration (in orange), and (2) the Euclidean distance between the level master problem optimal solution $\tilde{\mathbf{z}}^i$ in each iteration and the level master problem optimal solution $\tilde{\mathbf{z}}^{i-1}$ in the previous iteration (in blue). Note that the optimality gap between the initial lower bound $(\sum_{\omega\in\Omega} \pi_\omega E^*(\omega))$ and the initial upper bound $(\min\left\{E(\mathbf{z}^*_{\text{avg}}), \min\{E(\mathbf{z}^*(\omega)) : \omega \in \Omega\}\right\})$ is already quite small, less than 0.7%. The level bundle method improves the upper bound fast in the first few iterations. A substantial change happens in iteration 47 when the level master problem is infeasible, after which the lower bound γ^i_{low} is increased, the optimality gap is decreased, and the stability center $\hat{\mathbf{z}}^i$ is changed which decreases the Euclidean distances significantly. A similar change happens at iteration 124, when the algorithm stops because the optimality gap becomes less than the stopping tolerance $\varepsilon = 0.1\%$. During the execution of the level bundle algorithm, the average and standard deviation of the second-stage subproblem solving times are 366 seconds and 14.4 seconds, respectively. The average and standard deviation of the level master problem solving times are 3.6 seconds and 0.7 seconds, respectively. Thus, solving the large second-stage LPs takes much more time than solving the level master problem.

Next we compare the L-shaped method with the level bundle method for solving the LP relaxation of problem (4). Other algorithmic strategies such as independent Magnanti–Wong cuts and parallel solving of second-stage subproblems were enabled for both methods. Figure 17 shows the objective value $E(\tilde{\mathbf{z}}^i)$ of optimal



(a) Level bundle method bounds and optimality gap versus iteration number.



(b) Level bundle method Euclidean distances between master problem solution and stability center (orange) and between successive master problem solutions (blue).

Figure 16 **Progress results for level bundle method.**

solution $\tilde{\mathbf{z}}^i$ of the L-shaped master problem (LMP-Phase1) at each iteration i (as upper bound), and the optimal objective value of the L-shaped master problem (LMP-Phase1) at each iteration i (as lower bound). Two dashed reference lines are also shown that represent the initial upper bound $\min\left\{E(\mathbf{z}^*_{\text{avg}}), \min\{E(\mathbf{z}^*(\omega)) : \omega \in \Omega\}\right\}$ and the initial lower bound $\sum_{\omega\in\Omega} \pi_\omega E^*(\omega)$. Note that the L-shaped method could not improve the initial upper bound or the initial lower bound after 340 iterations (about 35 hours of running time). An iteration of the L-shaped method takes about the same amount of time as an iteration of the level bundle method, and thus Figures 16a and 17 can be compared.

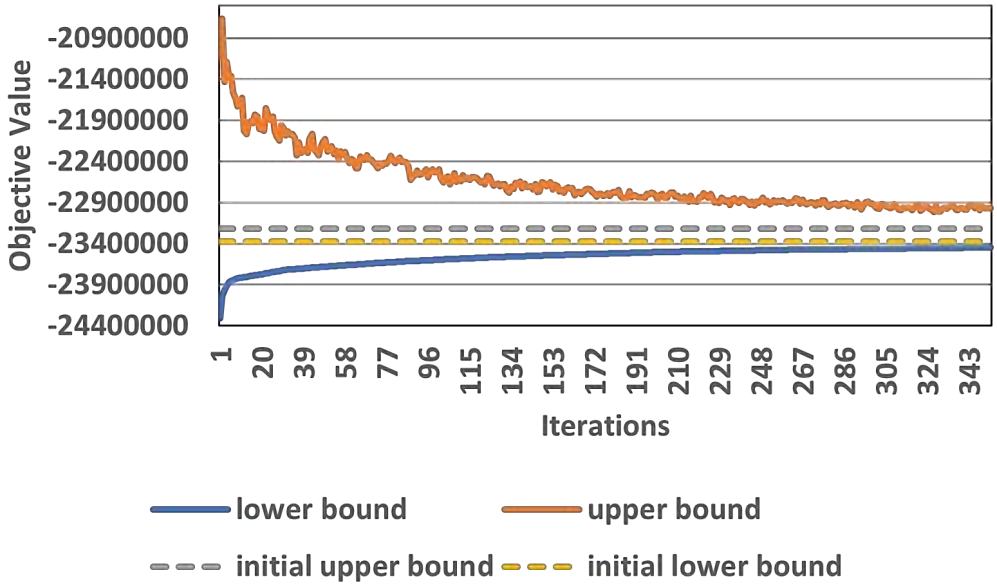


Figure 17 **L-shaped method bounds versus iteration number.**

E.1.3. Effect of Adding Independent Magnanti–Wong Cuts Next we present the effect of adding independent Magnanti–Wong cuts on the efficiency of solving the LP relaxation of problem (4) with the level bundle method. In each iteration i in which the level bundle master problem (LMP) is feasible, version MW-yes solves second-stage subproblem (SP)$(\bar{\mathbf{z}},\omega)$ for both $\bar{\mathbf{z}}=\tilde{\mathbf{z}}^i$ and for $\bar{\mathbf{z}}=\mathbf{z}^{i+1}_{\text{core}}$, and adds both cuts to constraints (8a) in the level bundle master problem (LMP), whereas version MW-no solves second-stage subproblem (SP)$(\bar{\mathbf{z}},\omega)$ for $\bar{\mathbf{z}}=\tilde{\mathbf{z}}^i$ only, and adds only one cut to constraints (8a) in the level bundle master problem (LMP). Both versions were used to solve the LP relaxation of problem (4) with the level bundle method, for 10 different randomly generated sets $\Omega_1,\ldots,\Omega_{10}$ of scenarios, each set Ω_n containing $|\Omega_n|=20$ scenarios. Table 5 shows the number of iterations $i_{\max}$ until convergence (with optimality tolerance $\varepsilon=0.1\%$), and the solving time t_{LP} (in hours), for both versions on each set of scenarios. On average, version MW-no required 113.5% more iterations and 45.9% more solving time for convergence than version MW-yes, and thus adding independent Magnanti–Wong cuts increased the efficiency of the level bundle method.

Table 5 Independent Magnanti–Wong effectiveness validation.

Ω		Ω_1	Ω_2	Ω_3	Ω_4	Ω_5	Ω_6	Ω_7	Ω_8	Ω_9	Ω_{10}
MW-yes	$i_{\max}$	124	111	116	119	139	127	110	128	129	122
	t_{LP}	14.0	12.6	13.7	14.5	16.8	13.9	12.7	15.2	15.6	14.2
MW-no	$i_{\max}$	246	238	251	243	288	291	245	271	283	259
	t_{LP}	22.1	19.0	20.1	19.5	23.7	20.9	18.5	22.2	23.0	19.9

E.1.4. Separate Cuts versus Aggregated Cuts An alternative to the separate cuts for different scenarios ω in constraints (20a) of the master problem RMP, the cuts can be aggregated over the scenarios ω. Next, we compare the effect of solving master problems with separate cuts for different scenarios ω versus solving master problems with cuts aggregated over the scenarios ω on the gaps between various upper and lower bounds. Both versions were used to solve problems for 10 different randomly generated sets $\Omega_1,\ldots,\Omega_{10}$ of scenarios, each set Ω_n containing $|\Omega_n|=20$ scenarios. Table 6 shows the gap $\left(\text{gap}^2_{\text{avg}}=\left[E(\mathbf{z}^{**}_{\text{avg}})-\gamma^0_{\text{low}}\right]/|\gamma^0_{\text{low}}|\right)$ between the objective value $E(\mathbf{z}^{**}_{\text{avg}})$ for problem (4) of the first-stage solution of the average-scenario problem and the LP relaxation of problem (4), the gap (gap^3) between the objective value of the best integer feasible solution found in 48 hours of computing by the branch-and-Benders-cut search with separate cuts or aggregated cuts and the lower bound given by problem (RMP), and the gap (gap^4) between the objective value of the best integer feasible solution found in 48 hours of computing by the branch-and-Benders-cut search with separate cuts or aggregated cuts and the LP relaxation of problem (4), for each set of scenarios.

E.1.5. Solving Single Scenario Problems Table 7 shows the optimality gap (gap^1) for each single-scenario and average-scenario problem after 1 hour of computing time. The optimality gaps are below 1% for all these problems after 1 hour of computing time, and thus the single-scenario and average-scenario problems are relatively easy to solve to near optimality. Table 7 also shows the gaps ($\text{gap}^2=\left[E(\mathbf{z}^{**}(\omega))-\gamma^0_{\text{low}}\right]/|\gamma^0_{\text{low}}|$ or $\text{gap}^2=\left[E(\mathbf{z}^{**}_{\text{avg}})-\gamma^0_{\text{low}}\right]/|\gamma^0_{\text{low}}|$) between the objective values for problem (4) ($E(\mathbf{z}^{**}(\omega))$ or $E(\mathbf{z}^{**}_{\text{avg}})$) of the first-stage solutions of the single-scenario and average-scenario problems

Table 6 Comparison of results using separate cuts versus aggregated cuts.

Ω		Ω_1	Ω_2	Ω_3	Ω_4	Ω_5	Ω_6	Ω_7	Ω_8	Ω_9	Ω_{10}
$\text{gap}^2_{\text{avg}}$		3.1%	3.5%	4.1%	3.0%	3.7%	3.9%	3.6%	3.7%	4.2%	3.2%
Separate cuts	gap^3	0.2%	0.4%	0.8%	0.2%	0.9%	0.3%	0.7%	0.1%	0.4%	0.7%
	gap^4	1.9%	2.6%	2.0%	1.7%	1.8%	2.3%	2.2%	2.0%	2.1%	2.5%
Aggregated cuts	gap^3	3.1%	3.5%	3.8%	3.0%	3.7%	3.9%	3.6%	3.7%	3.1%	3.2%
	gap^4	3.1%	3.5%	4.0%	3.0%	3.7%	3.9%	3.6%	3.7%	3.8%	3.2%

and the LP relaxation of problem (4). The solution $\mathbf{z}^{**}_{\text{avg}}$ for the average-scenario problem has the best objective value for problem (4), but most of the single-scenario solutions $\mathbf{z}^{**}(\omega)$ also have good objective values for problem (4), with 17 of the 20 single-scenario solutions having gap^2 less than 4%. The initial set $\tilde{\mathbf{Z}}^0 = \{\mathbf{z}^{**}(\omega_9), \mathbf{z}^{**}(\omega_{10}), \mathbf{z}^{**}(\omega_{12}), \mathbf{z}^{**}(\omega_{14}), \mathbf{z}^{**}(\omega_{15}), \mathbf{z}^{**}(\omega_{16}), \mathbf{z}^{**}_{\text{avg}}\}$ contains 6 near-optimal single-scenario solutions and the average-scenario solution. Initial constraints (20b) result in the values of 3903 out of 4381 first-stage variables being fixed at the start of the branch-and-Benders-cut procedure.

Table 7 Summary results for solving single scenario problems.

ω	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7
gap^1	0.94%	1.12%	0.92%	0.87%	0.86%	0.67%	0.70%
gap^2	4.51%	4.02%	4.09%	3.93%	3.91%	3.69%	3.75%
ω	ω_8	ω_9	ω_{10}	ω_{11}	ω_{12}	ω_{13}	ω_{14}
gap^1	0.47%	0.56%	0.54%	0.57%	0.38%	0.33%	0.46%
gap^2	3.71%	3.50%	3.55%	3.79%	3.50%	3.66%	3.60%
ω	ω_{15}	ω_{16}	ω_{17}	ω_{18}	ω_{19}	ω_{20}	ω_{avg}
gap^1	0.38%	0.41%	0.31%	0.44%	0.44%	0.73%	0.46%
gap^2	3.50%	3.48%	3.84%	3.82%	3.93%	3.50%	3.11%

E.2. Additional Insights from Solutions

E.2.1. Crossdocking Containers Processed by Airport For the first-stage solution produced by Algorithm 4, Table 8 shows for each airport $a \in A_c$ the total number $M'(a)$ of crossdocking containers processed by airport a, and the total number $M'_c(a)$ of crossdocking containers of each type $c \in C$ processed by airport a. The container crossdocking capacity $M(a)$ per day for each airport a is also shown.

Table 8 Numbers of containers making crossdocking connections at crossdocking airports.

$a \in A_c$	$M(a)$	$M'(a)$	$M'_{c_1}(a)$	$M'_{c_2}(a)$	$M'_{c_3}(a)$	$M'_{c_4}(a)$	$M'_{c_5}(a)$
a^*_1	500	162	1	159	1	1	0
a^*_2	240	74	3	71	0	0	0
a^+_3	50	6	0	6	0	0	0

E.2.2. Geographic Distribution of Crossdocking Capacity For the first-stage solution produced by Algorithm 4, Figure 18a shows the geographic distribution of crossdocking capacity on inbound flights to crossdocking airports $a \in A_c$, and Figure 18b shows the geographic distribution of crossdocking capacity on outbound flights from crossdocking airports $a \in A_c$. The darker lines represent flights with greater crossdocking capacity.

(a) Geographic distribution of crossdocking capacity on inbound flights.

(b) Geographic distribution of crossdocking capacity on outbound flights.

Figure 18 Geographic distribution of crossdocking capacity.

E.2.3. Origin-Destination Distribution of Shipment Flows Figure 19 shows the Lorenz curve of the cumulative proportion of total amount of shipment flows (total over paths for each origin-destination gateway pair, and averaged over different scenarios) against the cumulative proportion of origin-destination gateway pairs (sorted in non-increasing order of amount of shipment flows). It shows that the top 1% and top 10% of origin-destination gateway pairs constitute approximately 33% and 77% of all shipment flows, respectively. This can be compared with Figure 13a which shows the Lorenz curve for demand.

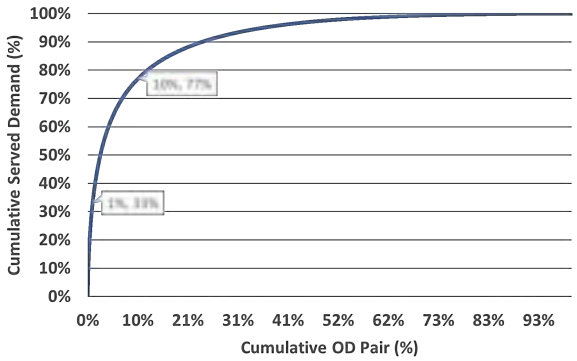


Figure 19 Lorenz curve of the cumulative proportion of total amount of shipment flows versus the cumulative proportion of origin-destination pairs.

E.2.4. Gateway-Airport Shipment Flows Table 9 shows the proportion p_1 of shipment flows between gateways and airports that go through the closest airport and the proportion p_2 of shipment flows between gateways and airports that go through the second closest airport, for different scenarios. Note that the proportion of shipment flows through the second closest airport is larger for scenarios with larger total shipment flows, suggesting that the relative use of the second closest airport increases as capacity at the closest airport becomes tighter.

Table 9 Shipment flow proportions between gateways and airports.

ω	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
p_1	91.1%	90.3%	90.7%	90.2%	89.8%	89.1%	89.4%	88.8%	89.4%	88.8%
p_2	8.9%	9.7%	9.3%	9.8%	10.2%	10.9%	10.6%	11.2%	10.6%	11.2%
ω	ω_{11}	ω_{12}	ω_{13}	ω_{14}	ω_{15}	ω_{16}	ω_{17}	ω_{18}	ω_{19}	ω_{20}
p_1	89.4%	88.8%	89.3%	88.8%	88.9%	89.0%	89.0%	88.9%	88.6%	88.6%
p_2	10.6%	11.2%	10.7%	11.2%	11.1%	11.0%	11.0%	11.1%	11.4%	11.4%

E.2.5. Flight and Crossdocking Container Utilization Table 10 shows the average flight capacity utilization u_1 among all self-operated flights, the average flight capacity utilization u_2 among all passenger flights that serve some demand in at least one demand scenario, and the average container capacity utilization u_3 among all crossdocking containers.

Table 10 Flight and crossdocking container utilization.

ω	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
u_1	39.8%	45.1%	45.6%	51.4%	50.5%	56.6%	57.2%	64.5%	62.1%	61.5%
u_2	21.7%	24.6%	24.6%	27.3%	27.2%	29.5%	29.8%	32.5%	31.8%	31.2%
u_3	75.7%	83.1%	83.0%	90.0%	88.4%	92.8%	94.1%	95.7%	95.5%	94.8%
ω	ω_{11}	ω_{12}	ω_{13}	ω_{14}	ω_{15}	ω_{16}	ω_{17}	ω_{18}	ω_{19}	ω_{20}
u_1	61.8%	66.1%	67.9%	67.3%	66.6%	67.7%	69.8%	70.6%	70.6%	75.0%
u_2	31.6%	34.0%	33.8%	34.1%	33.9%	33.8%	36.5%	36.6%	35.8%	40.2%
u_3	94.8%	95.7%	97.3%	97.1%	97.1%	97.4%	97.8%	98.6%	97.7%	99.2%

Figure 20 shows the composition of flight capacity utilization (averaged over different scenarios) on all self-operated flights. The flights were sorted in non-decreasing order of average flight capacity utilization. The average proportion of demand that is served through crossdocking connections, through en-route sorting connections, and by direct flights are shown in different colors.

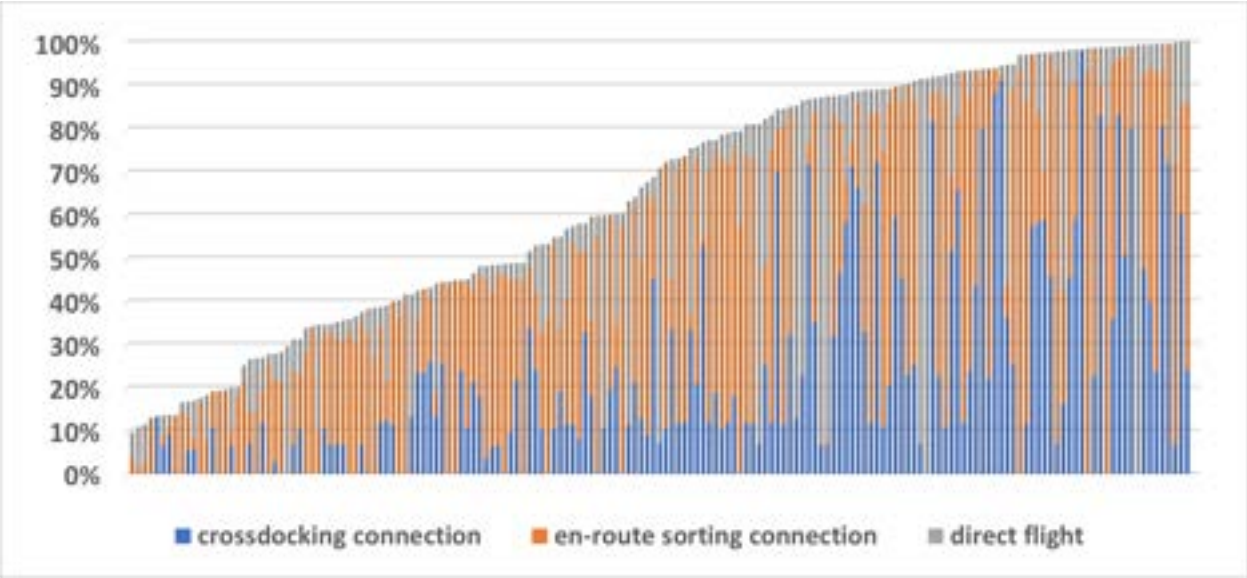


Figure 20 **Composition of flight capacity utilization.**