# Intra Market Optimization for Express Package Carriers

*Luke Schenk, Diego Klabjan*
*Department of Mechanical and Industrial Engineering, University of Illinois at Urbana-Champaign, Urbana, IL (lukeschenk@gmail.com, klabjan@uiuc.edu)*

The flow of packages and documents in collective groups, called splits, of an express package carrier consists of picking up the packages at customers' locations by a courier and bringing them to a station for sorting. Next the splits are transported either in bulk or containerized conveyances to a major regional sorting facility called the ramp. In this work we focus on the afternoon and evening operations concerned with stations and the ramp. We deal with the sorting decisions at the stations and the ramp, as well as the transportation decisions among these locations. We model these processes by means of a dynamic program where time periods represent time slices in the afternoon and evening. The resulting myopic problem is a linear mixed integer program. The overall model is solved by approximate dynamic programming where the value function is approximated by a linear function. Further strategies are developed to speed up the algorithm and decrease the time needed to find feasible solutions. The methodology is tested on several instances from an international express package carrier. Our solutions are substantially better than the current best practice.

## 1. Introduction

For overnight express package carriers, large volumes of packages and documents must be handled and often travel large distances in order to be delivered on time. In many cases, the time window between the pick up and delivery is less than 24 hours. Several operations and package movements must be completed within this time. These tasks can be divided into those that occur within the market where the package originated, and those outside the origin market. The intra market operations consist of a package being picked up by a courier from a customer, and any movement within its origin market until it reaches a location where it departs the origin market by plane or a different mode of transportation. Many carriers route packages on a hub and spoke network. After packages depart the origin market they travel to a hub before reaching the destination market and are ultimately received by the customer. In the U.S., major metropolitan areas constitute markets. All stations corresponding to surrounding towns and suburbs of a major city form a market. All packages from these towns and suburbs are gathered at a central facility called the ramp and are then transported to a hub.

It is important to route and sort packages within the origin market in such a manner that they meet their departure times dictated by schedules to prevent flight delays from propagating throughout the network. Customers use express shipment to ensure on-time delivery, so the failure of packages to depart the origin market can lead to poor customer service. The express package industry is also significant to the U.S. economy. In 1999, it was estimated that goods representing between 8.6% to 14.3% of the U.S. GDP were transported by express parcel shipments, Cannon (2000). While it is critical to deliver packages in a timely manner, excessive handling and package routing can lead to high operating costs for the carrier.

When packages are picked up by couriers, they are delivered to a nearby station. Next they are sorted and put into containers, which are transported to the ramp by conveyances, which include bulk and containerized trucks and vans as well as aircraft of various capacities. At the ramp the containers are broken down if required and the packages are sorted again. The ramp sorting process creates different containers, which are then loaded into an aircraft. This is a continuous time process throughout the afternoon and evening. The aircraft loading plans drive the entire process since they impose a requirement of a certain container type with specific packages at a given time. If packages are aggregated in a certain way at stations, they can bypass sorting at the ramp, which yields operating cost savings. Intra market package flow is depicted in Figure 1.

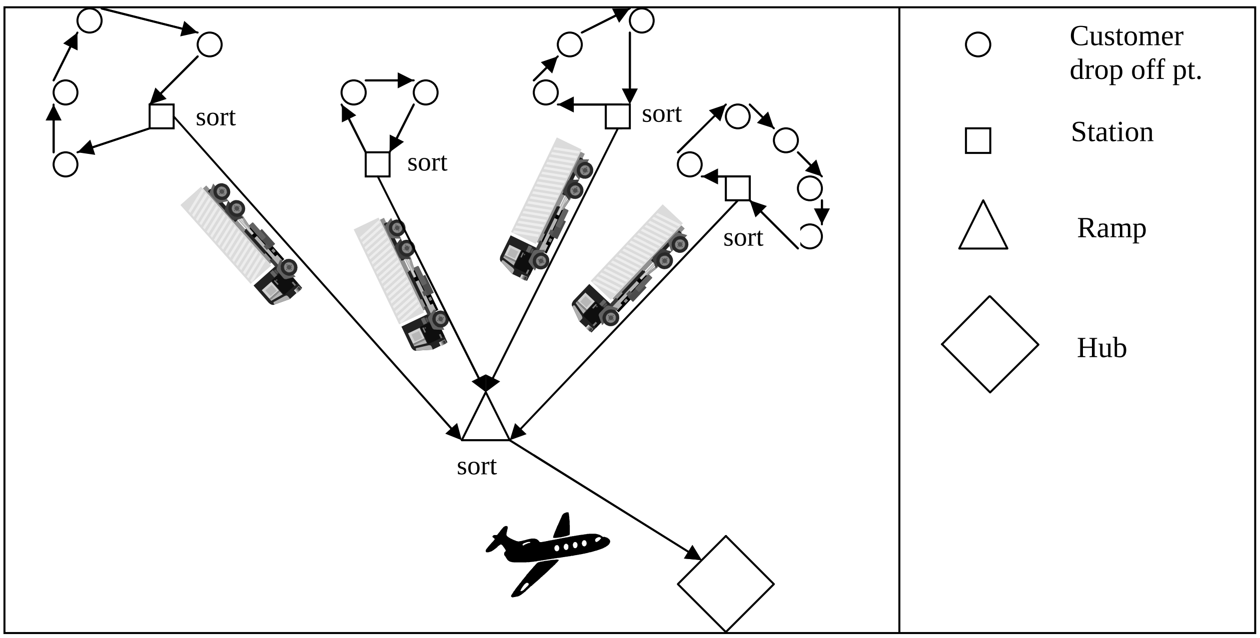


**Figure 1: Market Depiction**

The entire process has two key components: how to aggregate (sort) packages at stations in order to bypass sorting at the ramp, and what is the most cost effective transportation from the stations to the ramp. The output of the model is essentially a transportation schedule from the stations to the ramp with an indication of what kind of *splits*, aggregated groups of packages and documents, must be transported on each conveyance. We model the intra market afternoon or PM operations by means of a dynamic program (DP). We divide the entire time window into several smaller time periods, which yields a discrete time, finite time horizon DP. The resulting large-scale DP is solved approximately by using a linear function approximation to the value function.

The most important contributions of this work are a detailed model of the PM operations problem as well as the underlying solution methodology. While we use an existing algorithmic framework, we enhance and tailor it in several aspects. Results of our computational experiments show that linear approximations using gradient information (measuring the change in the value function with respect to the current state) generate good solutions to problems with many more time periods, longer travel times, and more constrained state and action spaces than other previously researched problems where this method was implemented. Since optimal solutions to the PM operations problem even for smaller markets are not known, our solutions are benchmarked against an integer programming formulation and the current operational practice. We also present strategies for choosing better initial values for the value function approximations and other methods to improve the speed and convergence of the algorithm. We improve the existing solutions in both direct cost and service level measured as the percentage of containers ready to depart the ramp on time. The former objective is captured by means of the one period cost function and the latter by properly forming the terminal conditions of the underlying dynamic program.

Section 2 discusses the PM operations problem in greater detail. Section 3 describes the dynamic programming formulation of the PM operations problem. Section 4 discusses details of the approximate dynamic programming algorithm and the corresponding solution methodology. Section 5 presents the results of computational experiments for the DP algorithm, a mixed integer linear programming formulation of the problem as well as comparisons with current operations in practice. We conclude the introduction with a literature review.

### *1.1. Literature Review*

Although there has been extensive research on vehicle routing and service network design, there is no work concerning the sorting and package handling details required for efficient intra market operations. The detailed modeling of package sorting and handling combined with routing decisions with a large number of discrete time periods presented in this work is unique. Due to the large number of time periods and multiple decisions concerning how to sort and route packages, our model is extremely large. A market with five stations, one ramp and over 300 time periods, which is considered a medium size market, resulted in a model with over 20 million decision variables and 3 million constraints when modeled as a simplified mixed integer program.

The related research follows two streams: the approximate DP area and service network design for express package carriers. We start with the latter. Kim et al. (1999) model multimodal express package delivery as a service network, which they call the Express Shipment Service Network Design (ESSND) problem. This paper presents solution methodologies for moving packages from their origins to their destinations with time windows for delivery. This work routes packages from the ramp to the hub while considering package transfer at an intermediate destination, and can similarly be used to route from the hub to the destination ramp on the outbound side. The volume of packages is considered constant to allow for flow conservation of aircraft to ensure that the solution can be repeated daily. For the PM operations problem, since conveyances only travel comparatively small distances within the market, we ignore repositioning and flow conservation and are able to capture fluctuations in package volume within a small time interval. The problem modeled here can be thought of as the input to the ESSND problem since the PM operations problem involves routing packages from stations to ramps and ESSND routes packages from ramps to the corresponding hub. Several problem reduction methods and heuristics are presented to decrease problem size without compromising model optimality. Further solution methodologies to the ESSND problem are given in Armacost et al. (2002). This paper introduces the notion of composite variables, which are essentially variables that capture both aircraft routing and package flow decisions. Thus there is no need for separate variables to represent these decisions, and the problem size and solution time are both reduced considerably. Both papers conclude that integer programming methods must be combined with heuristic strategies and other problem modifications to solve large problem instances.

We chose not to investigate these network reduction strategies since there are many more constraints in the PM operations problem compared to the ESSND problem and a large number of variables due to many time periods.. Instead, due to the natural discrete time component of the PM operations problem, we employ DP techniques. For problems modeled as a DP with many decisions and a large state space, the number of possible outcomes grows considerably and makes problems difficult to solve. These problems such as the PM operations problem with large state spaces require the use of approximate dynamic programming to provide an approximation to the value function. For a survey of approximate dynamic programming and work on other topics concerning the values of states including neuro-dynamic programming and Q-learning, the reader is referred to for example Bertsekas and Tsitsiklis (1996), Sutton and Barto (1998), and Powell and Van Roy (2004). Much of the recent work on dynamic programming applications and approximation algorithms to transportation problems is summarized in Powell et al. (2003) and Powell and Topaloglu (2003). Powell et al. (2001) provides a formal notation for the dynamic modeling of transportation problems and defines problem classes and terminology.

The seminal work on the topic is Powell et al. (1995). This paper provides a stochastic formulation of the dynamic assignment problem and approximates the problem for a continuous time and space setting as opposed to decomposing the problem into time periods. The model uses actual and forecasted demands to anticipate future events. Later work by Powell uses more of a standard dynamic programming methodology by decomposing the problem into time periods, solving the myopic problem for each time period, and using information from each time period for updating parameters of the approximate value function.

The work of Powell and Carvalho (1998) studies a dynamic fleet management problem and uses linear approximations based on gradients from the single time period problem in the value function and they present computational experiments that show that the results of their algorithm generates solutions that are within three percent of the LP relaxation of the integer programming formulation of the problem. Computational experiments show that the linear approximations work equally well for single and multi period travel times.

The differences in solving dynamic programs with single and multi period travel times are further discussed in the companion papers Godfrey and Powell (2002a and 2002b). They examine a stochastic version of the dynamic resource allocation problem where resources must be assigned to tasks over time and resources may be repositioned to a different location in order to fulfill the tasks. An efficient dynamic program which uses separable piecewise linear functions to approximate the value function is given that works well for single period travel times but degrades in performance for multi period travel times. Modifications are made to the nonlinear approximations for problems with multi period travel times that result in improved performance over the basic algorithm. The general problem with using separable piecewise linear functions in the case of multi period travel times is known as the long haul bias, which essentially results in repositioning a vehicle from a location that is farther away since this decision arises first. As a result, a higher cost is incurred to meet demand, resulting in poorer quality solutions. Further discussion of both linear and nonlinear value function approximations with multi period travel times is given in Topaloglu and Powell (2004).

The methods presented in this work differ from the work by Powell and his collaborators in the following important aspects. In our setting, the underlying optimization problem solved in each time period and iteration is a linear mixed integer program, which depends in a nonlinear manner on the given current state. In Powell's work, these problems are linear programs with linear value functions which yields a relatively easy way to compute gradients. As a result, the problems must be approximated by relaxing the integrality restrictions in order to measure the change in the value function with respect to the current state from the gradient.

# 2. Station and Ramp Operations

### *2.1. Suply*

On a given day, customers deposit packages at drop off points or couriers pick up packages from customer locations. Couriers then take these packages to facilities called *stations*. A station is a local facility that has the ability to sort and handle packages. Since we break up the time horizon, packages arrive at stations at fixed times, which are considered as incoming *supply points*. We assume that supply arrives at the beginning of a time period and volume departs a station at the end of a time period. We consider the overall volume of supply as opposed to the number of packages. Incoming supply can be differentiated based on destination, type of product such as hazardous or fragile, and package type such as a box or document. Each unique combination of these identifiers, referred to as *splits*, drives how volume will depart the station and eventually the market.

### *2.2. Sorting*

Once supply arrives to a station, it must be sorted to identify the volume by split. The *sorting belt* is essentially a conveyor belt that the supply is loaded onto and then moved within the facility. *Prioritized sorting* mimics a manual sorting procedure, where package handlers can pick and choose what packages to load into containers. Therefore splits that must depart the ramp earlier than others based on the demand profile can be sorted first. Based on a fixed speed of the conveyor belt or number of package handlers, there is a maximum amount of volume that can be sorted in a time period. So if the amount of incoming supply in a time period exceeds the sorting belt capacity, some of the supply will not be sorted in the time period in which it arrives and it must be determined how much of each split is actually sorted and how much is carried over to the following time periods. The reader is referred to Schenk (2005) for further extensions and complexities of the problem involved in practical situations. Examples of these extensions include a maximum capacity utilization per time period for each container, the sorting operations at each station and the ramp having unique start and end times, containers cannot be filled to the capacity, and so forth.

### *2.3. Demand Points*

At the ramp, there are fixed times when a specific split and container is needed to depart the market on time. These time, split, and container type (containers have different sizes) combinations are called *demand points*. Note that the container type of a demand point is given and is not a decision due to the desire of operational planning and other operational constraints (load balancing of outbound aircraft, loading congestion, etc.). We assume that the volume required by demand points is greater than the total supply, meaning that all supply can leave the market, i.e, demand exceeds supply.

To fulfill demand, a container with only one split can be created at a station and sent to the ramp to be directly used to meet a demand point. Such a container is known as a *pure container*.

Alternatively, a split can be combined with other splits at a station to form what is referred to as a *mixed container*. A mixed container must be *broken down* upon arriving at the ramp. This means that the volume is removed from the mixed container and re-sorted to separate the splits and eventually create pure containers that can then be used to fulfill demand points. The creation of a pure container may be impractical though, such as creating a pure container type that is not demanded, as this container would have to be broken down at the ramp.

### *2.4. Assigning Supply to Containers*

After determining how much of each split is sorted in a time period, we must decide where to put the sorted supply. A sorting belt has a number of locations where volume can be taken off the sorting belt as seen in Figure 2. As volume is being sorted at stations, the decision must be made as to what container type a split will be placed in and whether this will be a pure or mixed container. If there is sufficient volume, creating a pure container can be advantageous since mixed containers incur additional sorting and handling costs at the ramp. The process of re-sorting a mixed container also takes additional time which could result in supply missing its demand point, whereas a pure container can be immediately off-loaded at the ramp and used to meet demand. However, creating many pure containers can also lead to poor solutions as this can require additional conveyances in containerized or bulk form to transport the volume. Not to mention, creating a partial pure container may prevent a fuller container from being created at the ramp using mixed volume arriving from different stations.

### *2.5. Container Types and Load Positions*

Containers are divided into two main types, *refillable* and *non-refillable* containers. This division of containers is necessary due to how containers are filled with volume. Once a non-refillable container is closed, it cannot be reopened in order to add additional packages. A refillable container could be thought of as an actual conveyance where volume is loaded in bulk form. Packages can be added to refillable containers at any time. Thus there is a one-to-one correspondence between refillable containers and so-called bulk conveyances. On the other hand, several non-refillable containers can be loaded into a single conveyance. Another important difference between these container types is that non-refillable containers must be placed at *load positions* to obtain volume. Load positions are essentially slots/locations that are configured to hold specific sized containers while they are being loaded. Non-refillable containers also must be placed at load positions to properly close the container whenever it is filled up or ready to depart the station. For non-refillable containers, the number of currently loading containers is restricted by the number of containers of each type that can fit in load positions, see Figure 2. Also, at least one load position must be reserved for creating a mixed container. This is to prevent a condition where all load positions are occupied by pure containers and a split arrives that does not have a designated container at any load position. Figure 2 depicts sorting and package movements at a station. Throughout the time horizon, supply is sorted into refillable and/or non-refillable containers at stations. At some point in time, a container becomes ready to depart the station. This can be due to the container being filled to its capacity, or due to time constraints that require the container to depart the station to meet a demand requirement.

Containers come in different sizes. We address this by introducing container types. Within refillable and non-refillable containers we have several types, i.e., containers of different volume capacity.

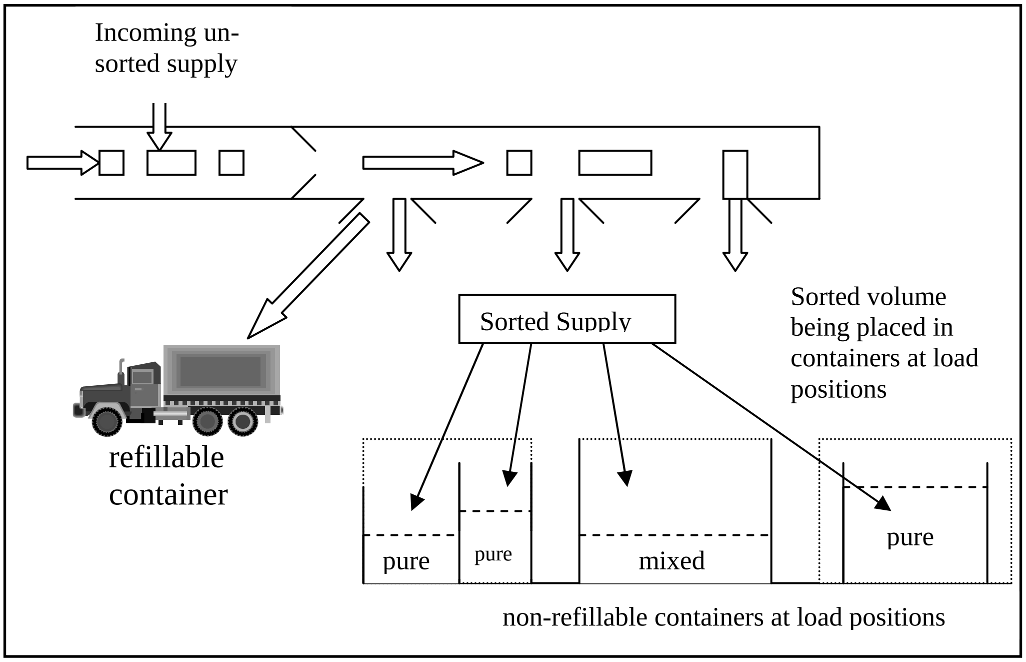


**Figure 2: Station Sorting Operations**

### *2.6. Container Conveyance Relationship*

Both refillable and non-refillable containers have a maximum capacity on volume. Containers are further distinguished by their *container footprint*. This parameter determines how load positions and conveyances are configured, in a sense taking into account the shape as well as the size of a container. For example, if a load position has a capacity of one, and a certain container type has a footprint of 0.5, two of these containers can be placed at the load position. The logic is similar for containers being placed on conveyances. So the number of containers that depart a station in a given time period times their footprint must be less than the footprint capacity of a conveyance being sent. It is assumed that any combination of containers can fit on a conveyance, provided that the total footprint capacity is equal to or less than the conveyance capacity.

### *2.7. Ramp Operations*

Ramps are typically much larger than stations and can sort more volume per time period. While non-refillable containers must still be placed at load positions to re-sort mixed volume, we do not consider a restriction on the number of load positions since there are often many of them. The decision as to what type of container to assign volume to at the ramp is also not considered since containers are placed at load positions based on the demand profile. It is also possible to have incoming supply that is picked up by couriers to go directly to the ramp. Clearly this supply must go on the sorting belt and be used to meet demand points that are not fulfilled by pure containers.

### *2.8. The Decision Making Process*

The main part of the input to the intra market problem is the supply and the demand profile. The supply profile gives the volume of each split that arrives at every station at a given time period while the demand profile represents the time, volume, and number of containers that are needed at the ramp and dictated by the departure schedules. In addition to these inputs, there are many more such as capacities, sort start times, footprints, load positions, etc. The main decisions to be made are the amount of each split to sort at each station and the ramp in each time period, which containers to assign the sorted splits to, and the resulting transportation plan.

The process is completely driven by the ramp. A day before the actual day of operations, the ramp engineer makes the aforementioned decisions and conveys them to station managers. In addition, the corresponding containers and conveyances are dispatched to the stations. Even though these are day to day operations, the entire problem is actually at the tactical level. The supply profile is very stable and therefore it is almost constant throughout a month. A single plan is made for a month and it is followed throughout the month. Month to month supply changes occur due to the seasonality of the express package business.

# 3. Dynamic Program

Initially we tried to solve the problem as a linear mixed integer program. However, even a simplified version of the model, which did not include all details of the problem and model constraints and handles only mixed containers, for the smallest instance was not able to find a solution with a reasonable gap. For larger instances the commercial integer programming solver was unable to find a feasible solution after several hours of computing time. Since there is an inherent time component in the model, we chose to model the problem as a dynamic program.

This section presents the details of the dynamic programming model. All notation, state and action space variables are given in this section.

### 3.1. Assumptions and Notation

The following basic notation is used throughout this paper.

| | |
|---|---|
| $F$ | Set of all stations (does not include the ramp) |
| $F \cup \{ra\}$ | Set of all locations (all stations plus the ramp) |
| $S$ | Set of all container types (unique container sizes) |
| $L$ | Set of all conveyance types (they differ by their capacity and cost, and potentially mode) |
| $A$ | Set of refillable container types |
| $B$ | Set of non-refillable container types, $S = A \cup B$ |
| $C$ | Set of volume possibilities in a container, $C = \{mix, pure\}$, where *mix* signifies a mixed volume container with more than one split and *pure* represents a pure volume container |
| $I$ | Set of all splits |
| $e_s$ | Footprint of each container $s \in S$ |
| $v_s$ | Volume capacity of each container $s \in S$ |
| $u_l$ | Footprint capacity of each conveyance $l \in L$ |
| $nblp_j$ | Number of load positions at station $j \in F$ |
| $cap_{tj}$ | The maximum amount of volume that can be sorted at station $j \in F \cup \{ra\}$ in time $t$ based on the sorting belt |
| $mf$ | The minimum amount of volume in a container before it is removed from a load |

position. It is based off of a proportion of the container capacity.

$w_{tij}$ The amount of split $i$ coming to station $j$ in time period $t$. These values determine the supply profile.

$d_{tis}$ The number of containers of type $s$ and split $i$ demanded in time period $t$. These values define the demand profile.

$g(j,ra)$ Number of time periods to travel from station $j \in F$ to the ramp

$f(s)$ Number of time periods needed to unload a container of type s

$sc$ The cost per unit of volume to sort and handle packages

$cc^{l}_{j,ra}$ The cost of shipping a conveyance of type $l$ from station $j \in F$ to the ramp. This parameter incorporates both fixed and variable transportation costs.

We also make the following assumptions.

- All supply comes into a station at the beginning of the time period.
- Containers are removed from load positions at the end of the time period. Any incoming supply that becomes ready during the time period can be removed at the end of the time period.
- Conveyances are sent at the end of time periods after containers are removed from load positions and they arrive at the beginning of a time period.
- Demand occurs at the end of a time period before conveyances arrive. For example, sorting may be performed during a time period and that volume can be used to meet demand in the same time period.
- At most one conveyance of a certain type is sent from a station to the ramp in each time period. This assumption enables us to capture the conveyance capacity and container footprint relationship precisely. In practice it never happens that more than one conveyance is sent in a time period since their capacity is much higher than the sort rate.
- We assume that it is not possible to create more than one refillable container of the same type in a time period. Due to the low sorting rate and high container capacity, this is for our instances without loss of generality.
- After a container of a certain type and split is removed from a load position, another container of the same type and split can be placed at a load position only after a one time period delay. In practice the load and unload time of a container is more than a single time period. This more general case is treated in Schenk and Klabjan (2006).
- Mixed containers that are ready to be sent to the ramp are set aside. We assume that if we choose to send one of these containers, then we must send all of them. This assumption is justifiable from the fact that typically only a few mixed containers are ready to leave. In most cases only a single one is ready. From the modeling point of view, if this assumption were not made, then we have to track the amount of every split in every single container (not just container type). In addition, the decision as to which mixed containers to send is a challenging optimization problem per se.

The sequence of events at a station and the ramp is depicted in Figure 3.

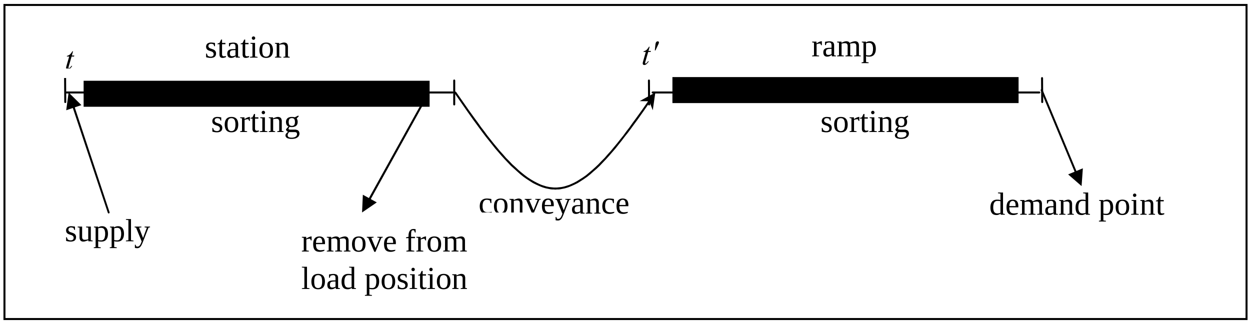


**Figure 3: Sequence of Events**

### *3.2. States*

From the point of incoming supply at stations, several resources must be tracked to ensure that all supply eventually departs the market and the decisions made in a time period are based on the appropriate resources that are present. The amount of volume in containers at load positions must be tracked to determine when containers are full and should be removed. Once removed, we need to know the number of available containers and amount of volume that could potentially depart the station in each time period. Since demand at the ramp is given in terms of containers rather than volume, we do not need to know the amount of pure volume ready to depart, simply the number of containers. For mixed containers, the volume of each split is important so we can guarantee that volume departs the station in order to be re-sorted at the ramp.

When the decision is made to have containers depart stations, we need to know the time that they depart the station as well as the time that they arrive at the ramp since this will not necessarily be the next time period due to multi period travel times. We let time $t$ be the time that containers or volume depart stations and let time $t'$ be the time that these assets arrive at the ramp. This information is then used to determine the amount of volume that must be re-sorted and the number of pure containers available to meet demand. Once supply is re-sorted, we need to track this amount to know what volume is available to potentially be used to meet demand.

It turns out that often we need to model the system dynamics for a pure container of type $i \in I$ as well as a mixed container. Therefore often we consider the split $i \in I \cup \{mix\}$, which is interpreted as a pure container of split $i$ if $i \in I$ or a mixed container if $i = mix$ in order to incorporate both pure and mixed containers across the system dynamics. Consider volume of split $i \in I$ in container type $s \in A \cup B$. This split can either be in a pure container or in a mixed container with other splits. We use $(s, pure)$ to indicate that we are referring to the amount of split $i \in I$ in a pure container of type $s \in A \cup B$ and $(s, mix)$ to represent split $i \in I$ being aggregated together with other splits in a mixed container of type $s \in A \cup B$. Figure 4 depicts the states, actions, and sorted volume at a station.

### *3.3. State Variables*

The following variables are used to represent the states of the system at the end of time period *t*.

| | |
|---|---|
| $\hat{r}^{s}_{tij}$ | The number of ready containers of type $s \in A \cup B$ of split $i \in I \cup \{mix\}$ at station $j \in F$ at time *t*. |
| $\bar{r}^{sc}_{tij}$ | Amount of split $i \in I$ type $c \in C$ at station $j \in F$ that is currently being loaded into a container of type $s \in A$ at a load position at time *t*. |
| $r^{sc}_{tij}$ | Amount of split $i \in I$ type $c \in C$ at station $j \in F$ that is in container type $s \in A$ at time *t* and is ready to be sent. |
| $\hat{r}^{s}_{tt^{'}i,ra}$ | The number of containers of split $i \in I$ type $s \in A \cup B$ that are sent to the ramp from any station at time *t* and arrive to the ramp at time $t^{'}$. |
| $\tilde{r}^{mix}_{tt^{'}i,ra}$ | The amount of mixed volume of split $i \in I$ that leaves any station at time *t* and arrives at the ramp at time $t^{'}$. |
| $r^{NS}_{tij}$ | Non sorted volume of split $i \in I$ at facility $j \in F \cup \{ra\}$ at time *t*. |
| $r^{MV}_{ti,ra}$ | Amount of split $i \in I$ that has already been sorted at the ramp but has not been used before time *t* to fill demand points. |
| $\hat{r}^{s}_{ti,ra}$ | The number of pure containers of split $i \in I$ type $s \in A \cup B$ that are at the ramp and have not been used before time *t* to fill demand points. |

To clarify the notation involving *c*, the quantity $r^{s,pure}_{tij}$ is the amount of split *i* at station *j* in time period *t* in all pure containers of type *s* that are ready to be sent. If this quantity is positive, it means that only split *i* is loaded in these containers. Similarly, $r^{s,mix}_{tij}$ represents the same quantity except that we consider all mixed containers of type *s* that are ready to be sent. In addition to split *i*, these containers may have other splits in them. We denote by $R_t$ the vector of all these state variables at time period *t*.

### *3.4. Action Variables*

The following action variables are used in the dynamic program.

| | |
|---|---|
| $\hat{y}_{tij}^{s}$ | Binary variable equal to 1 if container type $s \in A$ of split $i \in I \cup \{mix\}$ is removed from a load position at station $j \in F$ by the end of time period *t*. |
| $\hat{x}_{tij}^{ls}$ | Number of containers of type $s \in A \cup B$ of split $i \in I \cup \{mix\}$ that depart station $j \in F$ at time *t* on conveyance $l \in L$. |
| $z_{tij}^{sc}$ | Amount of split $i \in I$ at station $j \in F$ that at time *t* is sorted and assigned to a container of type $s \in A \cup B$, with either mixed (*c*=mixed) or pure (*c*=pure) volume, $c \in C$. |
| $\lambda_{tj}^{s}$ | Binary variable equal to 1 if all mixed containers of type $s \in A \cup B$ are sent from station $j \in F$ at time *t*. |
| $x_{tij}^{sc}$ | Amount of split $i \in I$, type $c \in C$ sent that departs station $j \in F$ at time *t* in container type $s \in A \cup B$. |
| $\hat{\varsigma}_{tj}^{l}$ | Number of conveyances of type $l \in L$ that leave station $j \in F$ for the ramp at time *t*. Based on our assumption this is a binary variable. |
| $\theta_{ti,ra}^{s}$ | Number of pure containers of type $s \in A \cup B$ of split $i \in I$ that are used in time period *t* to meet demand. |
| $\psi_{ti,ra}^{MV}$ | Amount of volume of split $i \in I$ that came to the ramp in mixed containers that is used to cover demand in time *t*. |
| $\sigma_{tij}^{s}$ | Binary variable equal to 1 if a non-refillable container $s \in A$ of split $i \in I \cup \{mix\}$ is occupying a load position at station $j \in F$ at time *t*. |
| $z_{tij}$ | Variable representing the prioritized amount of split $i \in I$ that is sorted in time period *t* at $j \in F \cup \{ra\}$ |

Let $X_t$ correspond to the vector of all these action variables at time *t*.

### *3.5. System Dynamics*

Figure 4 pictorially describes the parameters, state, and action variables representing station operations and departing conveyances, which are modeled by the following system dynamics. The number of containers that are ready to potentially depart is based on the number that are removed from load positions and those that depart the given station.

$$\hat{r}_{t+1,ij}^{s} = \hat{r}_{tij}^{s} + \hat{y}_{tij}^{s} - \sum_{l \in L} \hat{x}_{tij}^{ls} \quad s \in A, i \in I \cup \{mix\}, j \in F \qquad \textbf{Eq. 1}$$

To track the currently loading non-refillable containerized volume, we consider two options: removing the container from a load position or leaving it in a position. If it is not removed, the amount currently in a container at a load position plus any additional volume added to that container during the given time period must be captured, which is represented by the first term in Eq 2 and can be seen in Figure 4. If a container is removed, clearly the volume in the next time period is zero as shown in the second term. For mixed volume of split $i \in I$, we must denote that it is a mixed container being removed, as shown in Eq. 3.

$$\bar{r}_{t+1,ij}^{s,pure} = \min\left\{\bar{r}_{tij}^{s,pure} + z_{tij}^{s,pure}, v_s(1-\hat{y}_{tij}^{s})\right\} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 2}$$

$$\bar{r}^{s,mix}_{t+1,ij} = \min\left\{\bar{r}^{s,mix}_{tij} + z^{s,mix}_{tij}, v_s(1 - \hat{y}^{s}_{t,mix,j})\right\} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 3}$$

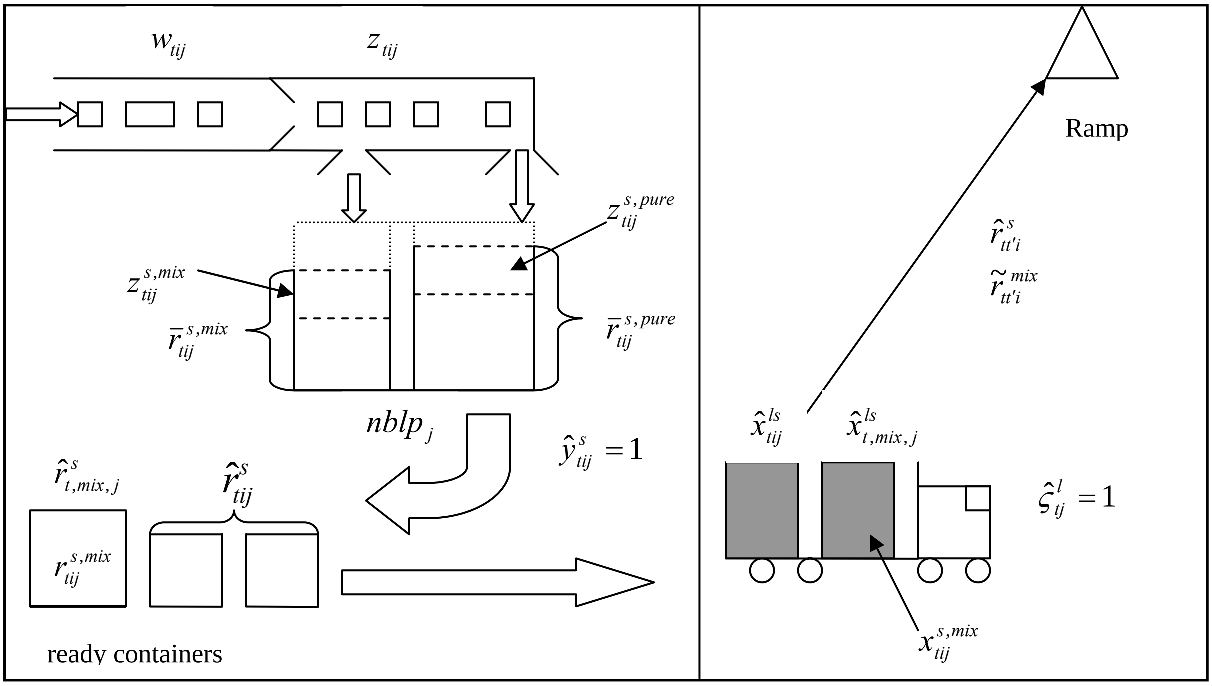


**Figure 4: The Actions and States at Station *j***

The following system dynamic for non-refillable containers accounts for the mixed volume that has been removed from load positions and could potentially leave the station. If a container is removed, then the amount of volume in the container is now ready to potentially leave.

$$r^{s,mix}_{t+1,ij} = r^{s,mix}_{tij} + \min\left\{\bar{r}^{s,mix}_{tij} + z^{s,mix}_{tij}, v_s \cdot \hat{y}^{s}_{t,mix,j}\right\} - x^{s,mix}_{tij} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 4}$$

Mixed refillable containerized volume is monitored in a similar fashion.

$$r^{s,mix}_{t+1,ij} = r^{s,mix}_{tij} + z^{s,mix}_{tij} - x^{s,mix}_{tij} \quad s \in B, i \in I, j \in F \qquad \textbf{Eq. 5}$$

The next system dynamic accounts for container arrivals at the ramp. The number of containers that arrive at the ramp at $t'$ is based on the number that depart stations if the departure time from that station plus the travel time to the ramp is equal to $t'$. Since the entire pure container is used to meet demand points, it is not necessary to track the amount of volume in such containers.

$$\hat{r}^{s}_{t+1,t'i} = \hat{r}^{s}_{tt'i} + \sum_{\substack{j \\ t+g(j,ra)=t'}} \sum_{l \in L} \hat{x}^{ls}_{tij} \quad s \in A \cup B, i \in I, t' > t \qquad \textbf{Eq. 6}$$

Since a mixed container must be broken down and re-sorted at the ramp, it is the volume of each split and not the actual number of containers that are of importance. Mixed volume will arrive at the ramp and be available to potentially be re-sorted at time $t'$ if the station departure time plus the travel time and the time to unload the volume based on the container type equals $t'$.

$$\tilde{r}^{mix}_{t+1,t',i,ra} = \tilde{r}^{mix}_{tt'i,ra} + \sum_{j \in F} \sum_{s \in A \cup B} \sum_{\substack{t \\ t+g(j,ra)+f(s)=t'}} x^{s,mix}_{tij} \quad i \in I, t' > t \qquad \textbf{Eq. 7}$$

The amount of non-sorted volume at the ramp is based on the direct incoming supply, mixed volume arriving from stations, and the amount that is sorted.

$$r^{NS}_{t+1,i,ra} = r^{NS}_{ti,ra} + w_{t,i,ra} + \tilde{r}^{mix}_{tti,ra} - z_{ti,ra} \quad i \in I \qquad \textbf{Eq. 8}$$

The amount of non-sorted volume at a station is based on the direct incoming supply and the amount that is sorted and assigned to a container type.

$$r^{NS}_{t+1,ij} = r^{NS}_{tij} + w_{tij} - \sum_{s \in A \cup B} \sum_{c \in C} z^{sc}_{tij} \quad i \in I, j \in F \qquad \textbf{Eq. 9}$$

The amount of volume that arrived to the ramp as mixed volume and has already been re-sorted is based on the amount that is sorted and any volume used to fulfill demand points.

$$r^{MV}_{t+1,i,ra} = r^{MV}_{ti,ra} + z_{ti,ra} - \psi^{MV}_{ti,ra} \quad i \in I \qquad \textbf{Eq. 10}$$

The number of ready pure containers available at the ramp is based on any pure containers arriving from stations at time *t* and those used in the current time period to fulfill demand points.

$$\hat{r}^{s}_{t+1,i,ra} = \hat{r}^{s}_{ti,ra} - \theta^{s}_{ti,ra} + \hat{r}^{s}_{tti,ra} \quad s \in A \cup B, i \in I \qquad \textbf{Eq. 11}$$

### *3.6. Action Space*

This section briefly describes some of the constraints in the action space. The remaining action space constraints can be found in the appendix.

The amount of supply that is assigned to a container must not exceed the amount that is sorted.

$$\sum_{s \in A \cup B} \sum_{c \in C} z^{sc}_{tij} \le z_{tij} \quad i \in I, j \in F \qquad \textbf{Eq. 12}$$

The following four constraints ensure that the amount of mixed or pure volume in a non-refillable or refillable container does not exceed the container capacity.

$$z^{s,pure}_{tij} \le v_s - \bar{r}^{s,pure}_{tij} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 13}$$

$$\sum_{i \in I} z^{s,mix}_{tij} \le v_s - \sum_{i \in I} \bar{r}^{s,mix}_{tij} \quad s \in A, j \in F \qquad \textbf{Eq. 14}$$

$$z^{s,pure}_{tij} \le v_s - r^{s,pure}_{tij} \quad s \in B, i \in I, j \in F \qquad \textbf{Eq. 15}$$

$$\sum_{i \in I} z^{s,mix}_{tij} \le v_s - \sum_{i \in I} r^{s,mix}_{tij} \quad s \in B, j \in F \qquad \textbf{Eq. 16}$$

The number of containers on a conveyance is bounded by the conveyance capacity. Note that this expression captures footprint capacity exactly only if $\hat{\varsigma}^{l}_{tj}$ is binary.

$$\sum_{i \in I \cup \{mix\}} \sum_{s \in A \cup B} e_s \cdot \hat{x}^{ls}_{tij} - u_l \cdot \hat{\varsigma}^{l}_{tj} \le 0 \quad l \in L, j \in F \qquad \textbf{Eq. 17}$$

The next two constraints relate pure containers and re-sorted mixed volume at the ramp to demand points. Demand points can be fulfilled by mixed or pure volume. The number of pure containers of type $s \in A \cup B$ used to meet demand for a split has to be less than or equal to the number of containers of that type and split demanded. Also, the overall volume used, which is given by the capacity of pure containers and the amount of re-sorted mixed volume, must be less than the overall volume demanded.

$$\theta_{ti,ra}^{s} \leq d_{tis} \quad s \in A \cup B, i \in I \qquad \textbf{Eq. 18}$$

$$\sum_{s \in A \cup B} v_s \cdot \theta_{ti,ra}^{s} + \psi_{ti,ra}^{MV} \leq \sum_{s \in A \cup B} v_s \cdot d_{tis} \quad i \in I \qquad \textbf{Eq. 19}$$

The following four constraints concern volume occupying load positions. To count the number of pure containers in load positions, Eq. 20 states that if any volume is assigned to, or already sorted into a container, the corresponding indicator variable $\sigma_{tij}^{s}$ must be equal to one. Constraint Eq. 21 keeps a similar count on mixed containers at load positions. If volume of any split is assigned to, or at a load position in a mixed container, $\sigma_{t,mix,j}^{s}$ must equal one. Constraint Eq. 22 guarantees that the number of occupied load positions times the corresponding container footprint is less than the total number of load positions at a station, while the last constraint reserves at least one load position for mixed volume. This is to allow the creation of a mixed container at all times, since there may be incoming supply of a split that does not have a pure container at a load position.

$$v_s \cdot \sigma_{tij}^{s} - z_{tij}^{s,pure} \geq \bar{r}_{tij}^{s,pure} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 20}$$

$$v_s \cdot \sigma_{t,mix,j}^{s} - \sum_{i \in I} z_{tij}^{s,mix} \geq \sum_{i \in I} \bar{r}_{tij}^{s,mix} \quad s \in A, j \in F \qquad \textbf{Eq. 21}$$

$$\sum_{i \in I \bigcup \{mix\}} \sum_{s \in A} e_s \cdot \sigma_{tij}^{s} \leq nblp_j \quad j \in F \qquad \textbf{Eq. 22}$$

$$\sum_{s \in A} \sigma_{t,mix,j}^{s} \geq 1 \quad j \in F \qquad \textbf{Eq. 23}$$

### *3.7. Cost Function*

The objective of the model is to minimize the costs associated with transporting all supply from stations to the ramp. There are also handling and sorting costs at stations, however this cost is constant based on the amount of supply. The first term in the cost expression Eq. 24 represents the sorting and handling cost at the ramp for mixed volume. The second term represents a conveyance departing station *j* for the ramp. The costs associated with this departure are a fixed cost to use the conveyance as well as mileage and driver costs based respectively on distance and time.

$$c_t(X_t, R_t) = sc \cdot \sum_{i \in I} \psi_{ti,ra}^{MV} + \sum_{l \in L} \sum_{j \in F} cc_{j,ra}^{l} \cdot \hat{\varsigma}_{tj}^{l} \qquad \textbf{Eq. 24}$$

The variables of the right hand side of the cost function are part of $X_t$. To summarize, the dynamic program consists of the cost function Eq. 24, system dynamics Eq. 1 – Eq. 11 and the action space is constrained by Eq. 12 – Eq. 23, Eq. 35, Eq. 36, and Eq. 38 – Eq. 50 given in the appendix. To incorporate missed deadlines and supply not leaving the ramp, a large penalty is given for having non zero state variables in the last time period.

# 4. Solution Methodology

The general methodology used to solve the dynamic program is based on the stochastic gradient algorithm for approximate dynamic programming, see Powell and Van Roy (2004) and Powell et al. (2003). Even though the dynamic programs developed in this work are deterministic, to comply with the existing terminology and literature on the subject, we call the presented algorithm the stochastic gradient algorithm. We use linear approximations of the value function for the stochastic gradient algorithm since this was shown to work well for problems with multi period travel times in Powell and Carvalho (1998). An additional benefit is computational tractability.

Let DP1 refer to the dynamic program given in Section 3 and let $V_t$ be the value of being in a particular state at time *t*. We first write the general form of the dynamic programming recursion Eq. 25 as follows.

$$V_t(R_t) = \min_{X_t}\left(c_t\left(X_t, R_t\right) + V_{t+1}\left(R_{t+1}(R_t, X_t)\right)\right) \qquad \textbf{Eq. 25}$$

Here we denote by $R_{t+1}\left(R_t, X_t\right)$ the transition function, which takes the current state and actions as arguments.

For our problem, the minimum is subject to constraints Eq. 12 – Eq. 23, Eq. 35, Eq. 36, and Eq. 38 – Eq. 50 and the initial value of all states is zero. Function $R_{t+1}\left(R_t, X_t\right)$ is given by equations Eq. 1 – Eq. 11 and the cost function is defined by Eq. 24. Since it is desired to have all supply depart the market, we want the final value of all states at the end of the time horizon to be zero also. Since we are minimizing the total cost, a large penalty is imposed for having non zero state values at time *T*, that is $V_T\left(R_T\right) = M$ for $R_T > 0$ and $V_T(0) = 0$ where *T* is the total number of time periods and *M* is a large number.

Standard discrete dynamic programming techniques can not be applied to the value function recursion due to the large state space and heavily constrained actions. So we attempt to replace the value function in the next time period $V_{t+1}(R_{t+1})$ by $\tilde{V}_{t+1}(R_{t+1})$. The approximate recursive formula, where $\hat{V}_t(R_t)$ is only a placeholder, is then given by the following equation.

$$\hat{V}_t(R_t) = \min_{X_t}\left(c_t\left(X_t, R_t\right) + \tilde{V}_{t+1}\left(R_{t+1}\left(R_t, X_t\right)\right)\right) \qquad \textbf{Eq. 26}$$

In order to have computational tractability, we selected a linear function approximation. This approximation is given by $\tilde{V}_t\left(R_t\right) = \sum_q \alpha_{tq} \cdot R_{tq}$ with $\alpha_{tq}$ being unknown parameters. The entire optimization problem now translates into the problem of finding the best possible values of $\alpha_{tq}$. Since there is a one-to-one correspondence among *α*'s and *R*'s, to simplify the exposition, we use the same notation for *α* as for *R*. For example, $\hat{\alpha}$ corresponds to the states denoted by $\hat{r}$ and $\tilde{\alpha}$ to the states denoted by $\tilde{r}$.

We are now faced with the decision of how to find good approximations for the values of states, or in other words, the values $\hat{V}_t(R_t)$. Instead of using only $\hat{V}_t(R_t)$, more information about the function is revealed by obtaining the gradient of $\hat{V}_t(R_t)$.

Given state $R_t$, it can be seen that Eq. 26 is a linear mixed integer optimization problem of the form

$$\bar{g}(R_t) = \min \bar{c}_t \cdot \bar{X}_t$$

$$A_t \bar{X}_t \le f(R_t) \qquad \textbf{Eq. 27}$$

$$\bar{X}_t \ge 0, \bar{X}_{ti} \text{ binary for selected } i.$$

Function $\bar{g}$ and $\hat{V}_t$ are virtually the same functions except that they are offset by a linear term in $R_t$. The rigorous argument about Eq. 26 being in the form of Eq. 27 can be found in Schenk (2005) and it uses the fact that $\tilde{V}_{t+1}(R_{t+1})$ is a linear function. For example, Eq. 2 has to be linearized and Eq. 36 needs to be modified to remove $R_t$ from the constraint matrix (as shown in the appendix). If not for these modifications to remove nonlinearities, $\bar{X}_t$ would be equal to $X_t$ and $A_t$ would simply be the constraint matrix corresponding to Eq. 12 – Eq. 23, Eq. 35, Eq. 36, and Eq.38 – Eq. 50. $\bar{X}_t$ contains the entire actions space variable $X_t$ and additional auxiliary variables. Note that the linear programming relaxation is solved to obtain the dual information which is then used to solve the mixed integer program to obtain decisions.

Let us discuss in more details the cost function $\bar{c}_t$. It clearly includes the right-hand side of Eq. 24. Consider the contribution of $\tilde{V}_{t+1}(R_{t+1})$ with respect to $\hat{r}$. We have

$$\sum_{s,i,j} \hat{\alpha}^s_{t+1,ij} \hat{r}^s_{t+1,ij} = \sum_{s,i,j} \hat{\alpha}^s_{t+1,ij} \left( \hat{r}^s_{tij} + \hat{y}^s_{tij} - \sum_{l \in L} \hat{x}^{ls}_{tij} \right) = \sum_{s,i,j} \hat{\alpha}^s_{t+1,ij} \hat{r}^s_{tij} + \sum_{s,i,j} \hat{\alpha}^s_{t+1,ij} \hat{y}^s_{tij} + \sum_{s,i,j} \sum_{l \in L} \hat{\alpha}^s_{t+1,ij} \hat{x}^{ls}_{tij} .$$

The first term of the right most expression does not depend on the decision variables and it thus contributes only toward the offset between $\bar{g}$ and $\hat{V}_t$. The remaining two terms involve decision variables and are thus part of $\bar{c}_t \cdot \bar{X}_t$. Similar expressions can be obtained for each state component.

Since obtaining a gradient of $\bar{g}$ is difficult due to the binary restrictions on variables, we instead approximate $\bar{g}$ by its linear programming relaxation $g$. Assuming that the functions $f$ and $g$ are differentiable, we define the gradient as follows, where $m$ represents the number of rows in the constraint matrix $A_t$ and $q$ is the coordinate index in $R_t$.

$$\Pi_{tq} = \frac{\partial g}{\partial R_{tq}} = \sum_{p=1}^{m} \pi_{tp} \cdot \frac{\partial f_p}{\partial R_{tq}} \qquad \textbf{Eq. 28}$$

In Eq. 28, $f = (f_1, ..., f_m)$ and $\pi_t$ is the optimal dual solution to Eq. 27. Even though function $g$ is not necessarily differentiable, we use Eq. 28 to approximate the descending direction since in our case $f$ is linear and thus differentiable.

Because the gradient can fluctuate, the following smoothing equation Eq. 29 is used to give weights to gradient information at the current iteration as well as previous approximations.

$$\alpha^n_t = (1 - \lambda^n) \cdot \alpha^{n-1}_t + \lambda^n \cdot \Pi^n_t \qquad \textbf{Eq. 29}$$

In the above equation, $\alpha_t^n$ is the value function approximation in iteration $n$ for state $r_t \in R_t$ at time $t$ and $\Pi_t^n = \left(\Pi_{tq}^n\right)_q$. The smoothing constant $\lambda^n \in (0,1)$ can be adjusted throughout the algorithm to help speed convergence.

### *4.1. Algorithm Summary*

Algorithm 1 gives the main steps of the approximate dynamic programming methodology. Note that in Step 1.2 we solve the mixed integer programming problem given by $\overline{g}$. Its solution dictates the next state, see Step 1.3. When solving this mixed integer program, we store the optimal dual values of its LP relaxation and use them to compute $\Pi_t^n$ based on Eq. 28.

---

**Step 0** *Initialization:* Choose an approximation $\tilde{V}_t^1$ for $V_t^1$ for all $t$. Set iteration counter $n$ = 1.

**Step 1** *Forward Pass:*

**Step 1.1** *Initialize forward pass:* Set $t$ = 1. Initialize $R_1$.

**Step 1.2** *Solve the myopic problem for a time period:* For time period $t$ solve the ap proximate myopic problem given by Eq. 26 by using the approximation $\tilde{V}_{t+1}^n$ subject to the action space constraints to get $X_t$.

**Step 1.3** *Apply the system dynamics:* Calculate $R_{t+1}$.

**Step 1.4** *Advance time:* Set $t = t + 1$. If $t \in T$ go to Step 1.2.

**Step 2** *Advance iteration counter:* Set $n = n + 1$.

**Step 3** *Value function update:* Calculate $\alpha_t^n = \left(1-\lambda^n\right)\cdot \alpha_t^{n-1} + \lambda^n \cdot \Pi_t^n$ for all $t \in T$ to provide an approximation $\tilde{V}_t^n$ for the next iteration. Go to Step 1.

---

**Algorithm 1: The Approximate Dynamic Programming Algorithm**

### *4.2. Updating Value Function Approximations for DP1*

This section gives updates to the value function approximations for a subset of the state variables. It shows what constraints in the myopic problem effect the perceived value of different state variables. Let $\pi^{12},\ldots\ldots,\pi^{23}$ be the dual variables corresponding to constraints Eq. 12 – Eq. 23 shown in Section 3 and $\pi^{35},\ldots\ldots,\pi^{49}$ be the dual variables for the action space constraints given in the appendix. The update equations obtained based on the gradients with respect to each approximating term as given by Eq. 28 for selected $\alpha_t$ are as follows.

$$\alpha_{tij}^{ns,mix} = \left(1-\lambda^n\right)\cdot \alpha_{tij}^{n-1,s,mix} + \lambda^n \cdot \left(-\pi_{tj}^{16s} + \pi_{tij}^{37s} + \pi_{tj}^{48s}\right) \quad s \in B, i \in I, j \in F \qquad \textbf{Eq. 30}$$

$$\alpha_{tij}^{ns,mix} = \left(1-\lambda^n\right)\cdot \alpha_{tij}^{n-1,s,mix} + \lambda^n \cdot \pi_{tij}^{37s} \quad s \in A, i \in I, j \in F \qquad \textbf{Eq. 31}$$

$$\tilde{\alpha}_{tti,ra}^{n,mix} = \left(1-\lambda^n\right)\cdot \tilde{\alpha}_{tti,ra}^{n-1,mix} + \lambda^n \cdot \left(\pi_{ti}^{40} + \alpha_{t+1,i,ra}^{n-1,NS}\right) \quad i \in I \qquad \textbf{Eq. 32}$$

$$\tilde{\alpha}_{tt'i,ra}^{n,mix} = \left(1-\lambda^n\right)\cdot \tilde{\alpha}_{tt'i,ra}^{n-1,mix} + \lambda^n \cdot \left(\sum_{t'>t} \tilde{\alpha}_{tt'i,ra}^{n-1,mix}\right) \quad i \in I, t' > t \qquad \textbf{Eq. 33}$$

Let us show next how to derive Eq. 30, which corresponds to $r_{tij}^{s,mix}$. Consider $s \in B, i \in I, j \in F$. The corresponding state $r_{tij}^{s,mix}$ appears on the right-hand sides of Eq. 16, Eq. 37, and Eq. 48 for these given *s,i,j*. It means that $\frac{\partial f_p}{\partial r_{tij}^{s,mix}} = 0$ for all *p* except three of them. For *p* corresponding to Eq. 16 we have $f_p(R_t) = v_s - \sum_{\bar{i} \in I} r_{t\bar{i}j}^{s,mix}$ and thus $\frac{\partial f_p}{\partial r_{tij}^{s,mix}} = -1$ from which we conclude $\pi_{tj}^{16s} \frac{\partial f_p}{\partial r_{tij}^{s,mix}} = -\pi_{tj}^{16s}$. We can similarly analyze Eq. 37 and Eq. 48. Based on Eq. 28 we obtain

$$\frac{\partial g}{\partial r_{tij}^{s,mix}} = \sum_{p=1}^{m} \pi_{tp} \cdot \frac{\partial f_p}{\partial r_{tij}^{s,mix}} = -\pi_{tj}^{16s} + \pi_{tij}^{37s} + \pi_{tj}^{48s},$$

which in turn justifies Eq. 30. The remaining equations Eq. 31-Eq. 33 can be verified in the same way.

### *4.3. Initial value function approximations*

To begin the algorithm, we must determine initial linear approximations to the value function. While these could theoretically be set to any value, the initial values as well as the weight given to data from previous iterations of the algorithm based on the smoothing factor can play a significant role in the performance of the algorithm. Basically, we are attempting to approximate the effect or value of having one additional unit of a particular resource.

Consider state $\hat{r}_{tij}^{s}$, which represents the number of containers of type *s* of split *i* that are ready to potentially depart station *j*. Since these containers must eventually be sent to the ramp, at some point in time they will be responsible for a fraction of the conveyance cost to transport the container to the ramp. If we estimate the cost of a container to be the fraction of space that the container occupies on the conveyance times the container footprint and consider the container departing on any possible conveyance, we obtain the following initial estimate for $\hat{\alpha}_{tij}^{s}$ given by Eq. 34.

$$\hat{\alpha}_{tij}^{1s} = \sum_{l \in L} \frac{e_s \cdot cc_{j,ra}^{l}}{u_l} \quad s \in A \cup B, i \in I, j \in F \qquad \textbf{Eq. 34}$$

As further examples of reasonable initial approximations consider the states $\tilde{r}_{tt'i}^{mix}$ and $r_{ti,ra}^{NS}$, which represent mixed volume traveling to the ramp and volume that has arrived at the ramp but has not yet been sorted. Clearly an additional unit of mixed volume would result in the extra cost incurred to re-sort mixed volume at the ramp, *sc*. Similar reasoning is used to estimate the initial values of other states to provide a better initial approximation than a trivial starting value such as zero.

### *4.4. Monotonicity Properties*

For some states, the sign of the initial approximation value $\alpha$ will affect the decisions made in the action space as well as future values of the approximating parameter. A necessary action such as volume leaving a station may be discouraged from happening due to the terms in the value function. Monotonicity properties of some of the state variables are important to determine the proper sign of some of the value function approximations and is used to properly update the approximations. We refer the reader to Schenk (2005) for proofs that the value function $V_t(R_t)$ is non-decreasing with respect to the following states.

1. $\hat{r}^s_{ti,ra} \quad s \in A \cup B, i \in I$
2. $\hat{r}^s_{tt'i} \quad s \in A \cup B, i \in I, t' \geq t$
3. $\hat{r}^s_{tij} \quad s \in A \cup B, i \in I, j \in F$
4. $\hat{r}^s_{t,mix,j} \quad s \in A \cup B, j \in F$
5. $r^{MV}_{ti,ra} \quad i \in I$

This information is used to restrict the approximation updates for these state variables to be nonnegative. For example, since $V_t(R_t)$ is non-decreasing in $\hat{r}^s_{ti,ra}$ , it makes sense to require that $\tilde{V}_t(R_t) = \sum_q \alpha_{tq} \cdot R_{tq}$ be non-decreasing in $\hat{r}^s_{ti,ra}$. This is equivalent to $\hat{\alpha}^s_{ti,ra} \geq 0$, which is imposed by modifying the updating formulas Eq. 30 – Eq. 33 for these states by applying the operator $(\cdot)^+ = \max(\cdot,0)$ on the right-hand side.

### *4.5. Improvement Strategies*

Several strategies are implemented to speed the algorithm. While gradient information should properly adjust the value function approximations, additional strategies and variable fixing is discussed in this section in order to accomplish this.

The first decision that must be made when considering the time based flow of supply from stations to the ramp is what container type to assign the incoming supply to. This decision is heavily based on the demand profile, as it is irrational to create a pure container of a type and split that is not demanded, or to create a pure container of a split that has a small amount of incoming supply at a station. Setting all initial approximations of the parameters corresponding to this decision to the same value would not convey this information in the myopic problems. So before the first initial pass of the algorithm, the amount of incoming supply at each station and the ramp demand profile are analyzed to determine good potential candidates for creating pure containers. Considerations as to the total number of pure containers made across all stations are compared with the demand profile to prevent creating more pure containers than the number demanded.

Naturally, whenever all incoming supply to a station has arrived and has been sorted, containers must be removed from load positions and conveyances must depart for the ramp as there is no benefit gained from having supply sit idle at a station if no more is to arrive. This forcing of supply out of stations is accounted for in the algorithm. Another reason supply may need to be forced out concerns departing the station in time to meet the last demand point for the given split and container type. If there is a pure container at a station and the current time plus the travel time from the station to the ramp and the container unload time equals the last demand point for the given split and container, this container must be removed and depart the station to have any chance of meeting demand.

The situation is similar for mixed containers. If there is mixed volume of any split at time *t*, and *t* plus the travel time, unload time, and the additional time to remove volume from the mixed container to be re-sorted equals the last demand point of any container type for that split, the container must leave the station immediately. This does not guarantee that the volume will meet the demand point due to the ramp sort capacity and other mixed volume still needing to be re-sorted. The ability to recognize volume missing demand points and the need to send mixed containers in an earlier time period (or more pure containers to lessen the queue at the ramp) will help alleviate the problem of late volume in later iterations of the algorithm. These additional strategies force the logical flow of supply along with insightful initial estimates for the value function approximations and provide sensible solutions after the first pass of the algorithm.

# 5. Computational Experiments

In this section we compare the DP model, an integer programming formulation given in Schenk (2005), as well as current operational practices. The models were tested on two different markets. Table 1 provides details about the range of the size of parameters for both market sizes, which will be referred to as M1 and M2 respectively. A time period of two minutes is used due to the need for timely actions for overnight delivery. There are other parameters that are not listed in this table that varied between the two market sizes. Both M1 and M2 are real world instances from U.S. geographic markets.

**Table 1: Market Parameters**

| | M1 | M2 |
|---|---|---|
| **Number of Stations** | 4-6 | 9-13 |
| **Number of Splits** | 22-26 | 22-26 |
| **Number of Container Types** | 5-7 | 7-9 |
| **Number of Conveyance Types** | 5-7 | 7-9 |
| **Number of Time Periods** | 250-350 | 350-500 |
| **Total Supply** | 14,000-18,000 cu-ft | 21,000-25,000 cu-ft |
| **Number of Action Variables in the Myopic Problem** | 20,000 | 45,000 |
| **Number of State Variables in the Myopic Problem** | 4,000 | 9,000 |
| **Number of Constraints in the Myopic Problem** | 7,000 | 15,000 |

Computational experiments were performed on a Dell personal computer with a Pentium® 4 1.80GHz processor, 1GB of RAM, and Windows XP operating system. Microsoft Visual C++ 6.0 was the development environment and we used ILOG CPLEX 8.0 with Concert Technology 1.0 as the mixed integer programming solver.

### *5.1. Small Market*

Table 2 gives solution characteristics for M1 using the three solution methods listed above. Due to proprietary reasons, we cannot discuss further details of the current operational practices, which we refer to as the baseline case.

**Table 2: M1 Solution Characteristics**

| | Objective | Run Time (hours) | Number of Conveyances | % Pure | Service Level | Conveyance Capacity Utilization | Container Capacity Utilization |
|---|---|---|---|---|---|---|---|
| **DP1** | 16.5% | 14 | 9 | 41.98 | 100% | 95.83% | 80.86% |
| **MIP** | -9.60% | 120 | 15 | 0 | 100% | - | - |
| **Current** | 0.00% | - | 11 | 57.21 | 99.90% | 97.20% | 61.49% |

Objective values are given as the percentage gap from the baseline solution. It can be seen that DP1 outperforms the current operations in practice by more than 16 percent. The mixed integer programming formulation (denoted as "MIP'') produces a worse solution than the baseline solution. This formulation was modified considerably in order to even obtain an integer solution, let alone one that is close to the LP relaxation value. Pure volume was not allowed as the addition of constraints and variables to incorporate pure volume and containers more than doubles the size of the model. Even with this greatly restricted mixed integer program, it took nearly 24 hours to find the first integer solution and after running it for three more days there was still a considerable IP/LP gap. If it were tractable to consider creating pure containers for MIP, this would most likely result in a better solution than current practice. However, it is unlikely that allowing pure containers in the MIP would lead to comparable or better solutions than the dynamic programs since the cost difference is primarily due to almost double the number of conveyances being used as opposed to just the additional sorting cost of mixed volume.

Comparing the current operations with DP1, we see that the dynamic program results in a lower cost because fewer conveyances are used. Although the current practice sends more pure containers and there is less volume to be re-sorted, it requires more conveyances and space is not as efficiently utilized. Current practice results in containers with less volume, thus causing wasted space and more conveyances to transport the volume. The container capacity utilization for both data sets is based off of the container capacity. Service level is the percentage of volume that is able to depart the market on time. As seen in Table 2, all volume is sorted in DP1 while a small amount of volume remains at the ramp in practice. Although more volume must be re-sorted at the ramp, it is sent early so that there is sufficient time to re-sort all of the mixed volume.

Figure 5 shows the objective value after each full pass of the dynamic programming algorithm for DP1. Clearly the initial approximations along with the strategy used to update the smoothing factor $\lambda^n$ play a factor in the convergence rate. It is important to note that the heuristics used to obtain initial approximations are based on the problem structure and not the specific data sets, therefore we firmly believe that the same successful behavior would be observed with different data sets. The strategy used for updating the smoothing factor $\lambda^n$ started with an initial value and then increased the value every given number of iterations by a certain amount to place a greater weight on the gradient information at later iterations of the algorithm. The number of iterations between two consecutive updates of $\lambda^n$ was selected in such a way as to keep the running time between two consecutive updates constant.

While the objective value for iteration 1 is at least 15 percent more costly than the final solution, this is actually a reasonable starting point as nearly all of the supply departs the ramp. Figure 6 gives the service level per iteration for DP1. Trivial initial approximations and not implementing additional rules such as forcing volume out of stations to meet demand points results in much poorer initial solutions and longer times to converge or even not finding reasonable solutions.

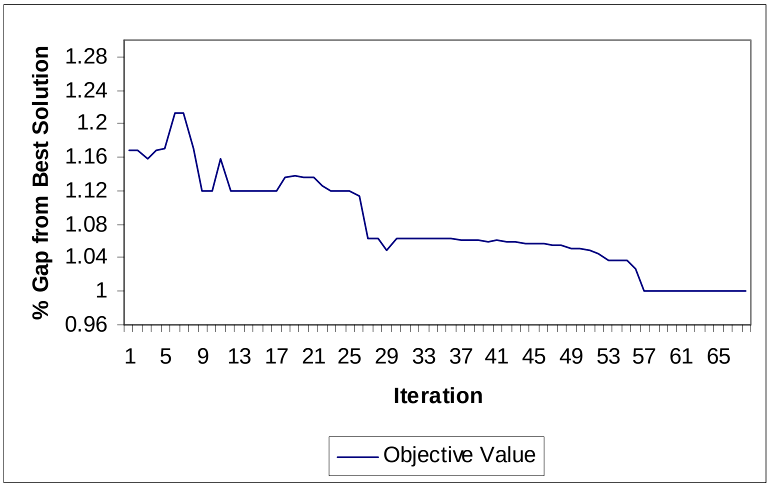


**Figure 5: Objective Improvements for Market M1**

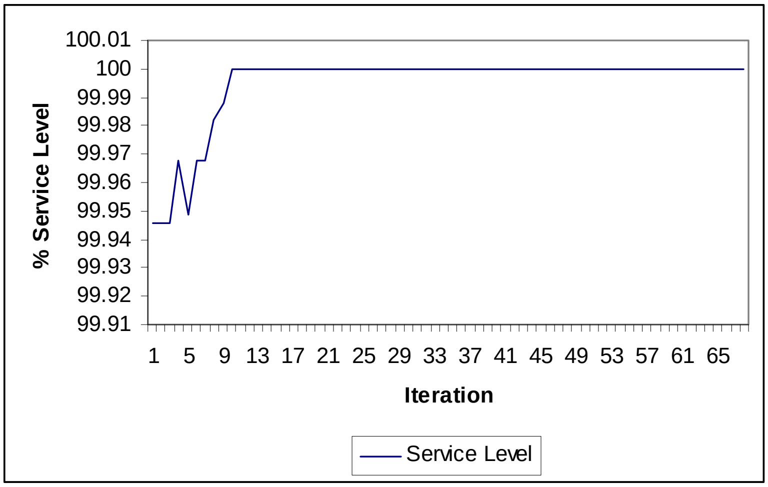


**Figure 6: Service Level Improvements for Market M1**

### *5.2. Larger Market*

Table 3 provides the results for the larger market. A feasible solution to even greatly reduced formulations from the mixed integer model could not be found. The dynamic programming solution significantly outperforms the current practice. The objective function gain is almost a third. Most of this gain comes from a significantly lower number of conveyances. Although the current practice sends slightly more volume in pure containers, this savings in sorting costs is dominated by the extra transportation cost of sending many more partially filled pure containers and there is still a significant amount of mixed volume that must be resorted. In addition to having a lower overall cost, the DP1 solution also has a larger service level and thus it dominates the baseline solution in both attributes.

**Table3: M2 Solution Characteristics**

| | Objective | Run Time (hours) | Number of Conveyances | % Pure Volume | Service Level | Conveyance Capacity Utilization | Container Capacity Utilization |
|---|---|---|---|---|---|---|---|
| **DP1** | 29.3% | 24 | 20 | 67.36 | 99.10% | 87.72% | 69.75% |
| **Current** | 0.00% | N/A | 32 | 71.66 | 98.90% | 100.00% | 38.72% |

As seen from the next to the last column, the baseline solution sends completely loaded conveyances. The conveyance capacity utilization of the DP1 solution is approximately 87%. On the other hand, many of the containers sent by the baseline solution are less than half full, while the DP1 solution sends containers that are more than two thirds filled. To summarize, the baseline solution sends fully loaded conveyances with containers filled up at approximately one third and the DP1 solution sends conveyances that are not completely loaded but the containers are filled to more than two thirds of their capacity.

Due to the problem size, run times were considerably longer for M2 compared with M1. Trials were stopped after 24 hours of run time for DP1 and the best solution is reported. While M2 is a larger data instance than M1, the main factor that causes an increase in run time is the number of time periods in the time horizon. As the number of time periods increases, the number of states and approximation values that must be generated increases dramatically due to states such as $\tilde{r}^{lsc}_{tt'ikj}$ where both the time $t$ when an action is taken and the time $t'$ when the effect of the action in time $t$ occurs must be accounted for. Figure 7 and Figure 8 give the objective value and service level per iteration for DP1 respectively with prioritized sorting at stations and the ramp. Although there are fewer iterations compared to the experiments with the smaller market M1, there is a clear improving trend in the objective value and service level. The initial value function approximations and forcing of supply out of stations again lead to reasonable initial solutions and service levels. Longer running times and further experimentation with updating the smoothing parameter could lead to improved solutions.

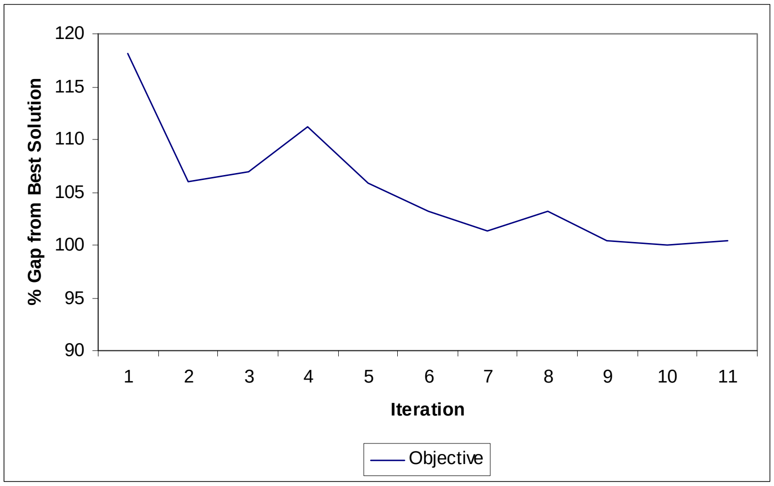


**Figure 7: Objective Improvements for Market M2**

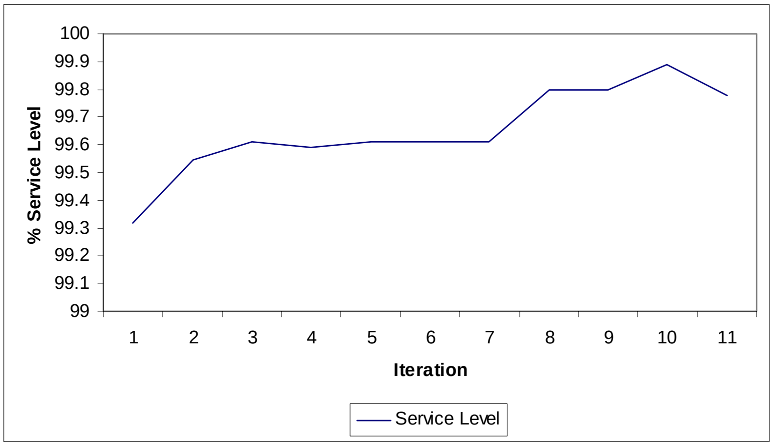


**Figure 8: Service Level Improvements for Market M2**

### *5.3. Analysis*

After comparing the models developed in this work across the two data sets, a few interesting trends emerge. An interesting point in comparing the solutions from the two markets is the amount of volume sent in pure containers rather than aggregated together as mixed volume. After analyzing the solutions, it appears that based on the given cost parameters, creating pure volume is only beneficial if this does not lead to excess cost from having to use additional conveyances. If there is not enough volume to fill a container close to the capacity, there is wasted space on the conveyance.

Consider the percentage of pure volume for both markets given in Table 2 and Table 3. Due to less supply for M1, the sorting belt at the ramp is able to process the roughly 60% of the volume that arrives as mixed volume. There is no need to ship more pure containers to lessen the queue at the ramp as all volume is able to be re-sorted on time, and creating more pure containers will likely lead to extra conveyances which would outweigh the savings of less volume to re-sort.

For the larger market M2, much more volume is sent in pure containers. There is simply too much incoming supply and many early demand points in the time horizon to send large amounts of mixed volume. Figure 9 gives a snapshot of the mixed volume that has arrived at the ramp but is not yet sorted and the demand points over this portion of the time horizon. This represents an early iteration of the approximate dynamic programming algorithm on data set M2. The plot starts with the beginning of the sorting operations at the ramp. There is initially a large queue of volume to be re-sorted due to volume from stations as well as direct incoming supply to the ramp. It can be seen that there is considerable congestion during time periods 125 through 135. While the demand points during this period represent only a subset of all splits and prioritized sorting leads to sorting those splits that are demanded during this period first, if a large amount of supply is sent as mixed volume to the ramp around this time, it may not be possible to re-sort all the supply to meet demand points.

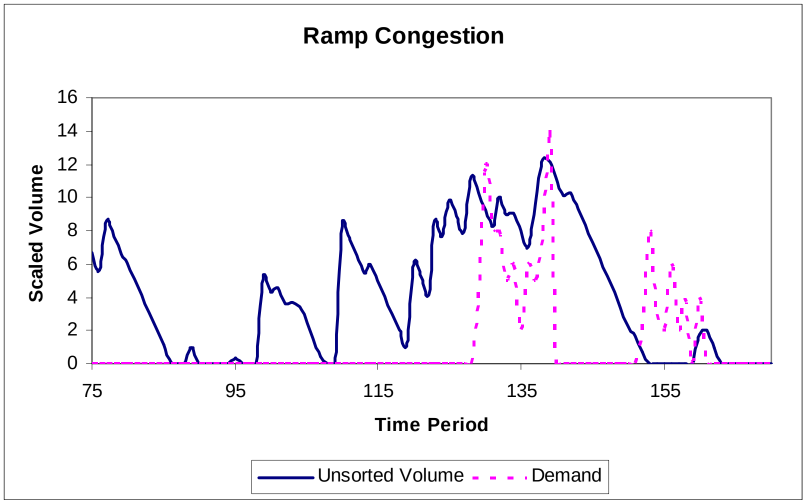


**Figure 9: Unsorted Ramp Volume and Demand**

So even if the creation of a new pure container led to an additional conveyance, this may be necessary in order to lessen the queue at the ramp and provide an acceptable service level. Due to the high conveyance and capacity utilization levels, it appears that the algorithm recognizes that when an additional conveyance is used, it should search for additional splits to make into pure containers to effectively use the conveyance capacity. Conveyance capacity utilization is lower for M2 compared with M1 due to containers being forced out of a station before the sort is complete in order to meet demand points.

# Acknowledgements

The authors would like to thank FedEx Express for providing the motivation for this work and the data used in computational experiments. We are much obliged to Byran D'Souza, Mike West, and Faisal Zaiman from FedEx Express. We are especially thankful to Mr. West for taking his time in extremely valuable conversations and touring with us.

# Appendix

This appendix gives the rest of the action space constraints that are not discussed in Section 3.

For mixed volume, it is not possible to ship one split while not shipping another since all the volume is mixed together. The next constraints enforce that either all or none of the mixed containers and mixed volume that are available to potentially depart are sent.

$$\sum_{l \in L} \hat{x}_{t,mix,j}^{ls} - \lambda_{tj}^{s} \cdot \hat{r}_{t,mix,j}^{s} = 0 \quad s \in A \cup B, j \in F \qquad \textbf{Eq. 35}$$

$$x_{tij}^{s,mix} - \lambda_{tj}^{s} \cdot r_{tij}^{s,mix} = 0 \quad s \in A \cup B, i \in I, j \in F \qquad \textbf{Eq. 36}$$

Note that these two families of constraints do not directly have the form stated in Eq. 27 since components $\hat{r}_{t,mix,j}^{s}$ and $r_{tij}^{s,mix}$ do not appear only on the right-hand side. To ensure these constraints are of the problem stated in Eq. 27, we transform Eq. 36 (the transformation for Eq. 35 is identical) into

$$x_{tij}^{s,mix} \le r_{tij}^{s,mix} \qquad s \in A \cup B, i \in I, j \in F \qquad \textbf{Eq. 37}$$

$$x_{tij}^{s,mix} \le M \cdot \lambda_{tj}^{s} \qquad s \in A \cup B, i \in I, j \in F,$$

where $M$ is a big number. This follows from the observation that $\min\left(M \cdot \lambda_{tj}^{s}, r_{tij}^{s,mix}\right) = \lambda_{tj}^{s} \cdot r_{tij}^{s,mix}$ since $\lambda$ is binary.

The following two constraints are needed to accurately track incoming supply. These constraints prevent assigning supply to a container in the same time period that a container is being removed. This also provides a delay of one time period for creating two containers of the same split and type.

$$v_s \cdot \hat{y}_{tij}^{s} + z_{tij}^{s,pure} \le v_s \quad s \in A \cup B, i \in I, j \in F \qquad \textbf{Eq. 38}$$

$$v_s \cdot \hat{y}_{t,mix,j}^{s} + \sum_{i \in I} z_{tij}^{s,mix} \le v_s \quad s \in A \cup B, j \in F \qquad \textbf{Eq. 39}$$

The amount sorted at the ramp is a decision variable to give precedence to splits with fast approaching demand points. This amount must be less than the unsorted volume currently at the ramp, which is represented by non sorted volume that arrived in previous time periods, direct incoming supply to the ramp, and mixed volume from stations. In addition to this constraint, the total amount of volume of any split sorted in a time period at the ramp is restricted by the ramp sort capacity.

$$z_{ti,ra} \le r_{ti,ra}^{NS} + w_{t,i,ra} + \tilde{r}_{tti,ra}^{mix} \quad i \in I \qquad \textbf{Eq. 40}$$

$$\sum_{i \in I} z_{ti,ra} \le cap_{t,ra} \qquad \textbf{Eq. 41}$$

In addition to the previously discussed constraints that model the actions that take place in a given time period, additional constraints are needed to ensure non-negativity of the state variables. The following constraints state that the number of containers departing a station, the number used to fulfill demand points at the ramp, and re-sorted mixed volume at the ramp must be less than the amount of containers and volume on hand.

$$\sum_{l \in L} \hat{x}_{tij}^{ls} - \hat{y}_{tij}^{s} \le \hat{r}_{tij}^{s} \quad s \in A \cup B, i \in I \cup \{mix\}, j \in F \qquad \textbf{Eq. 42}$$

$$\theta_{ti,ra}^{s} \le \hat{r}_{ti,ra}^{s} + \hat{r}_{tti,ra}^{s} \quad s \in A \cup B, i \in I \qquad \textbf{Eq. 43}$$

$$\psi_{ti,ra}^{MV} \le r_{ti,ra}^{MV} + z_{ti,ra} \quad i \in I \qquad \textbf{Eq. 44}$$

The following four constraints put a lower bound on the amount of volume that is in a container before it can be removed from a load position for $s \in A$ and before the container is ready to potentially leave for $s \in B$. These constraints are added to initialize the sorting of volume into containers in early iterations of the algorithm.

$$mf \cdot v_s \cdot \hat{y}_{tij}^{s} \le \bar{r}_{tij}^{s,pure} \qquad s \in A, i \in I, j \in F \qquad \textbf{Eq. 45}$$

$$mf \cdot v_s \cdot \hat{y}^s_{t,mix,j} \leq \sum_{i \in I} \overline{r}^{s,mix}_{tij} \quad s \in A, j \in F$$ **Eq. 46**

$$mf \cdot v_s \cdot \hat{y}^s_{tij} \leq r^{s,pure}_{tij} \quad s \in B, i \in I, j \in F$$ **Eq. 47**

$$mf \cdot v_s \cdot \hat{y}^s_{t,mix,j} \leq \sum_{i \in I} r^{s,mix}_{tij} \quad s \in B, j \in F$$ **Eq. 48**

If such a behavior is not imposed explicitly, then it is dynamically adjusted by the algorithm. In some circumstances, values *mf*s are part of the input (it captures the incentive of not handling too many containers). If this is not the case, we still add these constraints to speed up the convergence of the algorithm. In the latter case, we start with a large value that is gradually decreased.

Eq. 49 together with Eq. 12 imposes that the non sorted volume at stations $r^{NS}_{t+1,i,j}$ is nonnegative and Eq. 50 imposes the sort rate restriction at stations.

$$z_{tij} \leq r^{NS}_{tij} + w_{ijt} \quad i \in I, j \in F$$ **Eq. 49**

$$\sum_{i \in I} z_{tij} \leq cap_{jt} \quad j \in F$$ **Eq. 50**

# References


Armacost, A., Barnhart, C., Ware, K. (2002). Composite Variable Formulations for Express Shipment Service Network Design. *Transportation Science*, **36**, 1-20.

Bertsekas, D., Tsitsiklis, J. (1996). *Nuero-Dynamic Programming*. Athena Scientific, Belmont, MA.

Cannon P. (2000). *The Economic Significance of the Express Package Industry*. The W.J. Usery Jr. Center for the Workplace at Georgia State University.

Godfrey, G., Powell, W.B. (2002a). An Adaptive Dynamic Programming Algorithm for Stochastic Resource Allocation Problems, I: Single Period Travel Times. *Transportation Science*, **36**, 21-39.

Godfrey, G., Powell, W.B. (2002b). An Adaptive Dynamic Programming Algorithm for Dynamic Fleet Management, II: Multiperiod Travel Times. *Transportation Science*, **36**, 40-54.

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

Powell, W. B., Bouzaiene-Ayari, B., Simao, H.P. (2003). Dynamic Models for Freight Transportation. Available at http://www.castlelab.princeton.edu/Papers.

Powell, W.B., Carvalho, T. (1998). Dynamic Control of Logistics Queueing Networks for Large-Scale Fleet Management. *Transportation Science*, **32**, 90-109.

Powell, W.B., Carvalho, T., Godfrey, G., Simao, H.P. (1995) Dynamic Fleet Management as a Logistics Queueing Network. *Annals of Operations Research*, **6**, 165-188.

Powell, W.B., Shapiro, J. and Simao, H.P. (2001). A Representational Paradigm for Dynamic Resource Transformation Problems. In *Annals of Operations Research on Modeling* (C. Coullard, R. Fourer, and J. H. Owen, eds).

Powell, W.B., Topaloglu, H. (2003). Stochastic Programming in Transportation and Logistics. In *Handbooks in Operations Research and Management Science: Stochastic Programming* ( Shapiro, A., and Ruszczynski, A., eds.), Elsevier, Amsterdam.

Powell, W.B., Van Roy, B. (2004). Approximate Dynamic Programming for High Dimensional Resource Allocation Problems. In *Learning and Approximate Dynamic Programming: Scaling up to the Real World* (Si, J., Barto, A., Powell, W.B., Wunsch, D., eds.). John-Wiley and Sons, New York.

Schenk, L. (2005). *Intra Market Optimization for Express Package Carriers*. Master's thesis, Department of Mechanical and Industrial Engineering, University of Illinois at Urbana–Champaign.

Schenk, L., Klabjan, D. (2006). *Intra Market Optimization for Express Package Carriers with Station to Station Travel and Proportional Sorting*. Technical Report, University of Illinois at Urbana-Champaign. Available at http://netfiles.uiuc.edu/klabjan/www.

Sutton, R., Barto, A. (1998). *Reinforcement Learning*. The MIT Press, Cambridge, Massachusetts.

Topaloglu, H., Powell, W.B. (2006). Dynamic Programming Approximations for Stochastic, Time-Staged Integer Multicommodity Flow Problems. *INFORMS Journal on Computing*, **18**, 31-42.