Transportation Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

From Single Commodity to Multiattribute Models for Locomotive Optimization: A Comparison of Optimal Integer Programming and Approximate Dynamic Programming

Belgacem Bouzaiene-Ayari, Clark Cheng, Sourav Das, Ricardo Fiorillo, Warren B. Powell

To cite this article:
Belgacem Bouzaiene-Ayari, Clark Cheng, Sourav Das, Ricardo Fiorillo, Warren B. Powell (2014) From Single Commodity to Multiattribute Models for Locomotive Optimization: A Comparison of Optimal Integer Programming and Approximate Dynamic Programming. Transportation Science

Published online in Articles in Advance 28 Jul 2014

. http://dx.doi.org/10.1287/trsc.2014.0536

TRANSPORTATION SCIENCE
Articles in Advance, pp. 1–24
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.2014.0536

From Single Commodity to Multiattribute Models for Locomotive Optimization: A Comparison of Optimal Integer Programming and Approximate Dynamic Programming

Belgacem Bouzaiene-Ayari
Department of Operations Research and Financial Engineering, Princeton University, Princeton, New Jersey 08544, belgacem@princeton.edu

Clark Cheng, Sourav Das, Ricardo Fiorillo
Norfolk Southern Corporation, Atlanta, Georgia 30309
{clark.cheng@nscorp.com, sourav.das@nscorp.com, ricardo.fiorillo@nscorp.com}

Warren B. Powell
Department of Operations Research and Financial Engineering, Princeton University, Princeton, New Jersey 08544, powell@princeton.edu

We present a general optimization framework for locomotive models that captures different levels of detail, ranging from single and multicommodity flow models that can be solved using commercial integer programming solvers, to a much more detailed multiattribute model that we solve using approximate dynamic programming (ADP). Both models have been successfully implemented at Norfolk Southern for different planning applications. We use these models, presented using a common notational framework, to demonstrate the scope of different modeling and algorithmic strategies, all of which add value to the locomotive planning problem. We demonstrate how ADP can be used for both deterministic and stochastic models that capture locomotives and trains at a very high level of detail.

Keywords: approximate dynamic programming; locomotive planning
History: Received: September 2012; revision received: July 2013; accepted: November 2013. Published online in *Articles in Advance*.

1. Introduction

Locomotives represent a major investment for a railroad that might operate over 2,000 units, representing billions of dollars in assets. Railroads face the challenge of determining the right fleet size and mix, as well as operating the fleet in an efficient way. If the railroad maintains too small of a fleet, it will face the prospect of delayed trains, as well as additional costs from leasing locomotives. A fleet that is too large is expensive to own, maintain, and operate, and introduces the issue of storing additional locomotives in yards with limited space.

The locomotive planning problem is a complex and multidimensional problem with competing objectives. When assigning power to trains, planners have to take into consideration horsepower requirements (for speed), tractive effort (to get over steep grades), ownership, maintenance requirements (the locomotive may have to be routed toward a shop), and consist formation (locomotives are connected in sets of two to six locomotives, called *consists*, required to move a train).

Shop routing is a major complication. Locomotives are required by law to pass through maintenance shops every 92 days for routine maintenance, and from time to time, they need to be sent to the shop for mechanical repairs. If a locomotive misses a shop appointment, it has to be turned off and "dragged" to the shop, adding to the cost of moving a train and sometimes requiring dedicated movements (incurring additional crew costs). Railroads prefer to anticipate shop appointments, moving a locomotive toward a shop while maximizing its productivity. The goal is to arrive in time for a shop appointment, not too early and never late. At the same time, a particular shop may be heavily congested, and as a result the railroad has to balance the demands on different shops.

An important issue, widely ignored in prior research in locomotive optimization, is the presence of significant sources of uncertainty. Perhaps the most important is stochasticity in transit times, as well as the time required to process locomotives and trains through connecting yards. Compounding the problem is uncertainty in

the train schedule itself, as trains can be added to or dropped from the schedule with only a few days notice. Train additions are particularly common for commodities such as grain and coal.

Reflecting both its importance and complexity, locomotive optimization has attracted considerable attention from academics, consultants, and operations research groups within the railroads. Almost all of these prior models were deterministic, and developed during a period when commercial integer programming solvers were steadily maturing. The locomotive problem is not just a large integer program, it is a hard integer program that involves difficult features such as the need to bundle locomotives into consists, schedule locomotives into maintenance shops, manage foreign power, and deal with the need to occasionally delay trains.

This paper reports on a family of models that we refer to using the umbrella term PLASMA (Princeton Locomotive and Shop MAnagement system). These include the following:

PLASMA/SC. This is a single commodity flow model where all locomotives are represented as being the same. The goal of the model is to provide general guidelines of the following form: locomotives on inbound train A should be assigned to outbound trains B and C. This model balances flows on a network level, maintaining daily consistency patterns as well as first-in, first-out constraints. This model does not capture individual locomotives, and handles shop routing and the management of foreign power only at an aggregate level. Trains are not allowed to be delayed. The model produces a repeatable plan, with day-to-day consistency and matching inventories at the beginning and ending of the week.

PLASMA/MC. This is a multicommodity flow formulation that produces a deterministic integer programming model. Each locomotive is modeled individually so it can capture consist breakup costs, but locomotives are represented using a simplified set of attributes that allows them to be aggregated into four classes (commodities).

PLASMA/MA. This is a multiattribute resource allocation model that captures all of the attributes of a locomotive required to produce a truly realistic model. It also captures operational issues such as consist breakups, special equipment, returning foreign power to interchange points, and the particularly difficult task of routing power to maintenance shops. We have two flavors of this model: A strategic planning system used to determine fleet size and mix, and an operational planning model that works from a current snapshot of the current fleet.

PLASMA/SC and PLASMA/MC can both be solved using CPLEX. PLASMA/MC and PLASMA/MA can both be solved using approximate dynamic programming (ADP). This makes it possible to use PLASMA/MC as a benchmark between CPLEX and ADP. We note that both PLASMA/SC and PLASMA/MA are in production at Norfolk Southern, serving as evidence that both bring value to the planning process. In our experimental work, we show that CPLEX can solve PLASMA/MC, but only with several major simplifications, and even with these simplifications it is sharply limited in terms of the planning horizon.

PLASMA/MA can be used as a deterministic optimizer or as a stochastic optimization model that can handle variability in transit times, yard delays, locomotive failures, and changes to the train schedule. As of this writing, the deterministic version PLASMA/MA at Norfolk Southern is used in strategic planning to determine fleet size and mix. It is also currently being tested for tactical operational planning where it is used to plan locomotive movements over a one–five day horizon. This paper uses the stochastic optimization model to demonstrate for the first time that an optimization model can capture the uncertainty in transit times and yard delays, producing solutions that are significantly more robust than a deterministic model.

We faced our biggest challenge developing the strategic planning model that is used to recommend fleet size and mix. This requires that the model accurately capture locomotive productivity. We learned over the years that many of the standard simplifications that are accepted as part of the modeling process produced unacceptable results. For example, we started the model development by grouping locomotives into four major classes ("commodities") before finally making the transition to modeling each locomotive individually with over a dozen attributes. Consist breakups, shop routing, and the management of foreign power were all critical. We even found that arrival and departure times had to be modeled down to the minute; standard discrete time models did not work.

It is our belief from this research that ADP is uniquely well suited to handling high levels of detail, but is less skilled at managing a global vision of flows around the network *over time*. For example, ADP is not a good technology for producing repeatable solutions (as is handled with PLASMA/SC) from one day to the next, or ensuring that ending inventories match beginning inventories. ADP, however, can handle a high level of detail for each locomotive and train, globally optimizing over the entire network but only approximately optimizing over time.

The most difficult challenge we faced was calibrating the model to reproduce history. For this project, the most important metric was train delay. For example, for years the model produced delays that were too high, due almost entirely to problems with the data (primarily locomotives out of position, but sometimes

because of errors in the train schedule that made connections impossible). However, it was also possible to produce delays that were less than those observed in history. Although it is tempting to attribute instances of lower delays to the intelligence of the model, in practice they were always due to simplifications in the model. For example, early versions of the model did not handle shop routing and constraints on the formation of locomotive consists (such as the proper handling of foreign power and locomotive equipment types). Indeed, the railroads have learned that savings in locomotives from optimization models were due more to simplifications made in the model than the intelligence of optimization. A major finding of this project was that optimization was primarily a tool to mimic the intelligence of well-trained locomotive planners whose decisions, although perhaps not optimal, are better informed than those in any computer model.

This paper makes the following contributions. (1) We show that CPLEX has no difficulty solving the simplified PLASMA/SC model that is used to produce a high-level locomotive plan, but can only solve PLASMA/MC for limited horizons. (2) We present a multiattribute model that captures a very high level of detail, describe a solution strategy based on ADP and show, on a multicommodity formulation, that ADP produces a solution within 1% of optimalality. (3) We show that PLASMA/MA accurately captures fleet productivity when solved with ADP, producing overall train delays that are very close to history. (4) Along with finding the best consist with the right equipment, PLASMA/MA simultaneously manages the difficult problem of routing power to different shops while managing shop congestion. (5) Finally, we demonstrate for the first time that ADP can be used to solve a stochastic version of the locomotive assignment problem, producing a robust operating policy that performs well when tested under stochastic simulations.

Our presentation is designed to strike a balance between communicating the overall modeling and algorithmic strategy in a compact and clear formulation, while also representing the high level of detail required to capture real-world locomotive operations. We model locomotives and trains using multidimensional attribute vectors, which facilitates representing them at different levels of aggregation. The multiattribute model PLASMA/MA represents locomotives at three levels of aggregation within the same model.

Our use of ADP for this application required the development of new strategies for approximating value functions. For example, Simão et al. (2009) use value functions that are linear in the resource variable (drivers), and a hierarchical scheme to strike a balance between estimating the value of drivers at a high level of detail (which introduces statistical sampling problems) versus estimating drivers at a more aggregate level, which reduces statistical error but introduces bias. Topaloglu and Powell (2006) uses piecewise linear, separable value function approximations to model freight cars. Over the course of this project, we started with linear value function approximations (which were too unstable), progressed to piecewise linear, separable approximations, and finally ended up with a hierarchical approximation strategy that uses piecewise linear, separable value functions at two different levels of aggregation.

A theme of this paper is identifying the boundaries of classical integer programming technology using modern solvers. Our models start with PLASMA/SC, which is solved with CPLEX over a weeklong horizon. PLASMA/MC can be solved with CPLEX, but only over horizons up to around three days. ADP also uses CPLEX, but only to solve single-period locomotive assignment problems, which uses value function approximations to produce good solutions over time. The real value of CPLEX in PLASMA/MA was not in its ability to produce a higher quality solution, but rather in its consistency and reliability, which dramatically simplified model calibration.

The paper is organized as follows. Following the literature review in §2, §3 presents the modeling framework broken down into state variables, decision variables, exogenous information (for a stochastic model), transition function, and the objective function. The paper then presents three models. Section 4 presents a deterministic, single commodity formulation; §5 presents a deterministic, multicommodity formulation; and finally §6 presents an ADP algorithm for the multiattribute formulation. Section 7 describes how the shop routing problem is solved in an integrated way, illustrating how the ADP strategy can handle this important dimension of shop routing. Section 8 describes a series of experiments that demonstrate model validation against history, the stability of the model, the performance of the shop routing, and finally (for the first time) a demonstration that the model produces higher quality, more stable results for stochastic networks.

2. Literature Review

There is an extensive literature on optimization models for rail operations. These models range from single commodity models for managing generic fleets of containers (e.g., White 1972; and Herren 1977); to multicommodity models for handling multiple equipment types with substitution (Glickman and Sherali 1985; Dejax and Crainic 1987; Crainic and Rousseau 1988; Haghani 1989; Crainic and Laporte 1997; Cordeau, Toth, and Vigo 1998; Holmberg, Joborn, and Lundgren 1998; Joborn 2001; Lingaya et al. 2002 and Joborn et al. 2004). A separate line of research has focused on handling the high level of uncertainty in the demand for freight

cars (Mendiratta and Turnquist 1982; Jordan and Turnquist 1983); this research has continued under the general heading of "stochastic fleet management" or "dynamic vehicle allocation" (see the reviews in Powell, Jaillet, and Odoni 1995 and Powell, Bouzaiene-Ayari, and Simão 2006). This literature has laid a critical foundation for the locomotive optimization problem.

A separate literature has evolved around the more complex problem of managing locomotives. This problem has been modeled almost exclusively as a large-scale integer programming problem (see Cordeau, Toth, and Vigo 1998 for a review of the literature as of 1998; and Ahuja, Cunha, and Sahin 2005 and Nemani and Ahuja 2011 for more recent reviews). The complicating issue with locomotives is that it takes more than one locomotive to pull a train, and the minimum number of locomotives needed to pull a train is typically fractional (for example, a particular train may need 2.2 locomotives). Such problems are much harder than the "one resource, one task" problems that arise in container management problems.

There has been significant recent interest in models for locomotive optimization. Ziarati et al. (1999) describes the use of modern branch-and-cut integer programming algorithms for the locomotive problem, which was applied to the Canadian National Railway (Ziarati et al. 1997). Cordeau, Soumis, and Desrosiers (2000, 2001) apply Benders' decomposition to handle the simultaneous optimization of locomotives and cars. Ahuja et al. (2005) formulates the locomotive planning problem as a mixed-integer programming problem by modeling the flows of locomotives, and produced a plan that guides operations in the field, without reacting to real-time changes (similar to our PLASMA/SC model described below). This model considers issues such as "consist busting" (breaking costs of locomotives apart), the management of foreign power (locomotives owned by other railroads), and the handling of special equipment such as cab signals. The authors found, however, that execution times using the version of CPLEX available could exceed 10 hours. In Vaidyanathan and Ahuja (2008), a model based on flows of *consists* was developed that reduced consist breakups to almost zero; this formulation was also compared to a hybrid model for validation. This model, developed with CSX, also incorporated details such as communications and the routing of foreign power. Vaidyanathan, Ahuja, and Orlin (2008) formulate the locomotive routing problem to produce implementable tours for individual locomotives that considers refueling and maintenance. This is an extremely large integer programming problem that is solved using an aggregation/disaggregation technique.

3. A Modeling Framework

In this section, we provide a flexible modeling framework for the locomotive planning problem. This model provides the foundation for the single commodity, multicommodity, and multiattribute models. An effort has been made to strike a balance between providing the structure of the model at a high level, while still communicating the many details that were required to produce an accurate representation of locomotive operations. Most of the details of the models are provided in the appendices.

We present a single notational system where locomotives are represented using a vector of attributes. This system can be quickly adapted to produce single commodity, multicommodity, and multiattribute models. This representation has proven useful in other models of freight transportation systems (see in particular Powell, Shapiro, and Simão 2001, 2002; and Powell 2007, Chapter 12). We use the modeling framework of ADP, which can be quickly adapted to both deterministic models, where we optimize over an entire horizon, or the simulation-based strategy of ADP. This framework divides the model along five dimensions: the state variables, the decision variables, exogenous information (needed for the stochastic model), the transition function, and the objective function.

We present our model by assuming that decisions are made in discrete time steps $t = 0, 1, \ldots, T$. We are able to assign a locomotive to at most one train at a time, which limits the length of the time steps to four hours, which is roughly the length of the shortest train segment. For example, in the noon to 4 P.M. segment, we might assign a locomotive to a train that departs at 12:37 P.M. and arrives at a nearby yard at 3:15 P.M. with a planned departure at 3:45 P.M.; we will not assign a locomotive to the train departing from the second yard until we solve the 4 P.M. to 8 P.M. block, although we can still model the departure as happening at 3:30 P.M. It is important to emphasize, however, that although we may make decisions in four-hour blocks, all events are modeled in continuous time. So, we might decide at 8 A.M. that a locomotive that first becomes available at 9:32 A.M. should be assigned to a train that was available to move at 8:45 A.M., thereby implying a 47 minute delay.

3.1. The State Variable

For the locomotive optimization problem, the state variable captures the status of the locomotives and trains at a point in time. This is described using the following:

a = The vector of attributes describing a locomotive.
$\mathcal{A}$ = The set of all possible attribute vectors.
R_{ta} = The number of locomotives with attribute $a \in \mathcal{A}$ in the system at time t.
$R_t = (R_{ta})_{a \in \mathcal{A}}$.
b = The vector of attributes describing a train departure.
$\mathcal{B}$ = The set of all possible train attributes.

D_{tb} = The number of trains of type $b \in \mathcal{B}$ in the system at time t.
$D_t = (D_{tb})_{b \in \mathcal{B}}$.
$S_t = (R_t, D_t)$.

A detailed summary of the 14 locomotive attributes a and 23 train attributes b are given in Appendix A. Examples of locomotive attributes include current (or inbound) location, time of arrival, locomotive ID (used in consist breakup logic), owner, locomotive type, axles, horsepower, current train assignment, shop due date, maintenance date, and special equipment (cab signal, flush toilet, speed limiter). Examples of train attributes include origin (if en route), destination (or current location), train ID (used in consist breakup logic), estimated time of arrival, train type (e.g., merchandise, coal, grain, unit train).

For our most detailed operational model, the attributes of a locomotive include the following elements: the arrival time (or time of availability) of a locomotive, the next location (if moving) or current location (if at a yard), the estimated time of arrival (if the locomotive is currently moving), the owning railroad, locomotive special equipment (such as the *locomotive speed limiter* (LSL), the *cab signaling system* (CSS), and the *flush toilet system* (FTS)), the number of axles (a measure of the tractive effort of a locomotive, which determines its ability to move from a standstill or move up a steep grade), the horsepower (which determines the maximum speed of the locomotive), the ID of the current or previous train that it is currently assigned to (this is used to determine when locomotives are joined together in a consist), the next time it is due in a maintenance shop, and the shop repair status. A full description of the attribute vectors a and b are given in Appendix A.

Standard modeling practice for deterministic models has been to discretize time and then assume that all arrivals and departures occur at these time steps, as illustrated in Figure 1(a). Our work found that this did not provide an acceptable level of accuracy even for our simplest models, since one train might arrive at 12:37 P.M., a second might arrive at 1:12 P.M., and we might need locomotives from both trains to move a train that was originally scheduled to leave at 12:50 P.M., but which now has to leave after 1:42 P.M.

To handle such situations, we used a task-graph representation as depicted in Figure 1(b), where every train movement is represented as a specific arc unique to the movement. This representation allows us to model train departure times precisely, including the need to delay trains small amounts because of a late incoming train. Contrast this with an arc that simply moves between points in space time, common in multicommodity flow models (see Ahuja, Magnanti, and Orlin 1993). The task-graph representation allows a locomotive from train 2 to be assigned to train 3, requiring that the departure time of train 3 be delayed. We limit how much a train might be delayed, but the ability to handle train delays was found to be an important feature. In all of our models, we are able to represent arrival and departure times for locomotives and trains in continuous time.

The attribute vector notation simplifies the transition from the streamlined single and multicommodity models, to the more detailed models used for fleet sizing and operational planning. The vector b is assumed to always describe a single train. However, the attribute vector a may not uniquely identify a locomotive (particularly when a reduced attribute vector is used), and as a result R_{ta} may be greater than one. In fact, the ADP model represents locomotives at three different levels of detail at different points in time within the model. By contrast, the integer programming models use the same attribute vector throughout the planning horizon, but we can create models of varying computational complexity by changing the list of attributes that we wish to consider.

From the state vector R_t, we can compute two statistics that are used below to compute important costs. These are the following:

$\mathcal{F}$ = Set of foreign railroads.
R^F_{tf} = The number of foreign locomotives owned by railroad $f \in \mathcal{F}$ at time t, for which the railroad has to pay leasing costs. This is computed by summing the elements of R_{ta} where the ownership attribute of a corresponds to a railroad $f \in \mathcal{F}$.

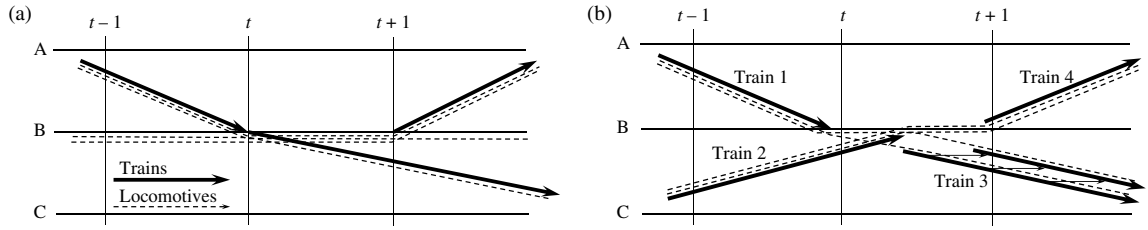

Figure 1 **Illustration of a Classical Time-Space Network for Multicommodity Flow Problems (a), and a Task-Graph Network That Provides for Continuous Arrival and Departure Times (b)**

R^{Shp}_{ts} = Number of locomotives in shop s at time t (in the shop, or waiting in line to enter the shop).

To model consist formulation, we introduce the following notation:

$\mathcal{I}_t$ = The set of train IDs at time t, where a train ID is represented using $i \in \mathcal{I}_t$.

$\mathcal{K}_{ti}$ = Set of locomotives that are identified with train ID $i \in \mathcal{I}_t$ at time t. We can think of the elements of $\mathcal{K}_{ti}$ as a set of train IDs or, equivalently, a set of attribute vectors $a \in \mathcal{A}$ where $R_{ta} = 1$.

If there are four locomotives in $\mathcal{K}_{ti}$ for some train ID i, and these are assigned to three different trains (two locomotives to one train, and one each to two separate trains), then this implies that we have "broken" the consist two times, each of which incurs a cost.

3.2. The Decision Variables

The locomotives available at time $t \in \{0, \ldots, T\}$ are acted on with different types of decisions, which we model using the following:

$\mathcal{D}^D$ = The set of decisions used to act on a train, consisting of "*move*" and "*hold*." These decisions are always used in the context of a unique train identified by its attribute vector b.

$\mathcal{D}^B$ = The set of decisions to assign a locomotive to a train in $\mathcal{B}$, where $d \in \mathcal{D}^B$ corresponds to a train $b_d \in \mathcal{B}$. Note that assigning a locomotive to a train is distinct from the decision to move the train itself.

d^H = The decision to hold a locomotive (the symbol refers to the fact that we are doing nothing to the locomotive).

d^{Set} = Decision to set off a locomotive that is currently attached to a train.

d^{Shp} = Decision to move a locomotive into a shop for repair or maintenance. This decision can only be used when a locomotive is in a yard where the shop is located.

$\mathcal{D}^R$ = The set of all decisions that can be used to act on a locomotive,
$= \mathcal{D}^B \cup d^H \cup d^{Set} \cup d^{Shp}$.

Each type of decision in $d \in \mathcal{D}^R$ acts on locomotives with an attribute vector in $\mathcal{A}$. We represent decisions using the following:

x^R_{tad} = The number of locomotives (resources), with attribute $a \in \mathcal{A}$, that we act on with decision $d \in \mathcal{D}^R$.

$x^R_t = (x^R_{tad})_{a \in \mathcal{A},\, d \in \mathcal{D}^R}$.

x^D_{tbd} = The number of trains, with attribute $b \in \mathcal{B}$, that we act on with a decision $d \in \mathcal{D}^D$. Since b always identifies a single train, $x^D_{tbd} \in \{0, 1\}$.

$x^D_t = (x^D_{tbd})_{d \in \mathcal{D}^D,\, b \in \mathcal{B}}$.

$x_t = (x^R_t, x^D_t)$.

We represent the feasible region for x_t using the following:

$\mathcal{X}_t$ = The feasible region for x_t that captures conservation of flow, integrality, and nonnegativity, given what we know at time t.

The equations that define $\mathcal{X}_t$ are given in Appendix C.

There are other decision variables that are determined directly by x^R_t and x^D_t through various constraints, which are described in Appendix C. For this reason, all of these variables are a function of x_t (we do not write this explicitly for notational compactness). These are the following:

w_{tb} = The delay imposed on train $b \in \mathcal{B}$ at time t due to the specific assignment of individual locomotives.

y^E_{tb} = The number of power units added on a train $b \in \mathcal{B}$ at time t for repositioning.

y^G_{tb} = The number of power units assigned to a train $b \in \mathcal{B}$ at time t to fulfill the goal requirement.

z_{tb} = The number of locomotive consist splits at time t, for the consist identified by train with attribute b. For example, if a four-locomotive consist is broken into a consist with two locomotives, along with two individual locomotives, this would produce $z_{tb} = 2$.

3.3. Exogenous Information

Exogenous information processes arise only in stochastic models. This information arises to describe transit time and yard delays, as well as changes to the train schedule. We model these changes using the following:

$\hat{R}_{ta}$ = Exogenous changes to locomotives with attribute a (primarily delays in transit times and yard delays) from information arriving between $t-1$ and t.

$\hat{R}_t = (\hat{R}_{ta})_{a \in \mathcal{A}}$.

$\hat{D}_{tb}$ = Exogenous changes to a train with attribute b.

$\hat{D}_t = (\hat{D}_{tb})_{b \in \mathcal{B}}$.

W_t = The exogenous information that arrived during time interval t.
$= (\hat{R}_t, \hat{D}_t)$.

We treat W_t as the information that first becomes known at time t (it is useful to think of W_t as information that arrives continuously between $t-1$ and t), and captures changes in arrival times (relative to the schedule) and yard delays, the addition of new trains to the schedule (or random train cancellations), and equipment failures (the unexpected requirement that a locomotive be returned to the shop for maintenance).

A delay in a train arrival is captured as a change in the estimated time of arrival (an attribute of the locomotives pulling the train). Changes in the train schedule are modeled as increases (schedule additions) or decreases (cancellations) to the vector $D_t = (D_{tb})_{b \in \mathcal{B}}$.

In our deterministic model, there is no exogenous information about locomotives and trains. For this reason, $\hat{R}_t = 0$ and $\hat{D}_t = 0$. The vector R_t is determined

purely from the starting state R_0 and the decisions we make that act on locomotives. In a deterministic model that represents the entire problem over time, D_0 would represent the entire train schedule. When we use ADP, we still model the schedule as if it arrives over time, even though there is no uncertainty. Thus, when we use ADP, D_t can be thought of as the trains that we are *thinking about* at time t, which generally refers to train departures over a fairly short horizon.

3.4. The Transition Function

The most natural way to model the evolution of locomotives and trains is through the use of *resource transition functions* that work at the level of individual locomotives and trains. We represent these using

$$a_{t+1} = a^M(a_t, d_t, W_{t+1}), \quad \forall (a_t, d_t) \in \mathscr{A} \times \mathscr{D}^R, \tag{1}$$

$$b_{t+1} = b^M(b_t, d_t, W_{t+1}), \quad \forall (b_t, d_t) \in \mathscr{B} \times \mathscr{D}^D. \tag{2}$$

Here, a_t and b_t are locomotive and train attribute vectors at time t, d_t is a type of decision acting on a locomotive or train at time t, and W_{t+1} would be (in a stochastic model) new information arriving between t and $t+1$. The attribute vectors a_{t+1} and b_{t+1} represent the future locomotive and the future train resulting from the implementation of decision d_t, given what we know at time $t+1$. In the case where d_t is a hold decision, a_{t+1} simply represents the same location at the next decision period. When d_t represents a train movement, a number of rules are applied that determine the precise departure time given all of the locomotives assigned to the train. For the stochastic model, we can capture variations in transit times that were not known when we made the decision at time t.

Our modeling strategy is relatively independent of the rules that govern how the locomotives and trains evolve over time. In the deterministic models, the attribute transition function performs the same role that the A matrix does in mathematical programming. In the ADP model (deterministic or stochastic), we implement the dynamics of a transition just as we would in any simulation model.

We can use the transition functions $a^M(\cdot)$ and $b^M(\cdot)$ to create matrices $\Delta^R_{t+1}(\omega)$ and $\Delta^D_{t+1}(\omega)$ that allow us to write the transition functions for R_t and D_t. Let

$$\delta^R_{t+1,a',ad}(\omega) = \begin{cases} 1 & \text{if } a' = a^M(a_t, d_t, W_{t+1}(\omega)), \\ 0 & \text{otherwise,} \end{cases}$$

$$\delta^D_{t+1,b',bd}(\omega) = \begin{cases} 1 & \text{if } b' = b^M(b_t, d_t, W_{t+1}(\omega)), \\ 0 & \text{otherwise,} \end{cases}$$

when we are following sample path ω and observe $W_{t+1}(\omega)$. We then let $\delta^R_{t+1,a',ad}(\omega)$ be the element of $\Delta^a_{t+1}(\omega)$ for row a' and column (ad). We form $\Delta^b_{t+1}(\omega)$ in a similar way. We drop the argument ω and view these matrices as random at time t, where we simulate them to compute the state of the system at time $t+1$. This allows us to write our transition equations using

$$R_{t+1} = \Delta^R_{t+1} x_t + \hat{R}_{t+1}, \tag{3}$$

$$D_{t+1} = \Delta^D_{t+1} x_t + \hat{D}_{t+1}. \tag{4}$$

Written this way, the random vector $\hat{R}_{t+1}$ would only represent exogenous arrivals to or departures from the system. An exogenous arrival might be locomotives coming from another railroad, or newly purchased/leased locomotives entering the system for the first time.

We are going to find it useful to introduce the concept of a *post-decision* state variable for locomotives. Let

$$a^d_t = a^M(a_t, d_t, W_{t,t+1}),$$

where $W_{t,t+1}$ is a forecast, made at time t, of the exogenous information arriving between t and $t+1$. We can interpret a^d_t as the attribute vector that we would *expect* the locomotive to have after being acted on using decision d. The most important illustration of these ideas for our work arises with travel times and the estimated time of arrival. If we send a locomotive from yard A to yard B, a^d_t would have yard B as the inbound location, and the time of availability (estimated time of arrival) reflects the scheduled time of arrival.

Using the forecasts $W_{t,t+1}$, we can create a forecasted indicator variable $\delta^R_{t,t+1,a',ad}$ and a forecasted incidence matrix $\Delta^R_{t,t+1}$. From this, we define the post-decision resource vector R^x_t given by

$$R^x_t = \Delta^R_{t,t+1} x_t. \tag{5}$$

We could then let

$$R_{t+1} = R^x_t + \hat{R}_{t+1},$$

but now $\hat{R}_{t+1}$ is capturing not just exogenous arrivals of locomotives but also updates due to random travel times. We note that $\hat{R}_{t+1}$ may be positive or negative.

In some cases, it is useful to use a general transition function $S_{t+1} = S^M(S_t, x_t, W_{t+1})$, where the decision vector x_t captures actions on all of the locomotives and trains. All of these representations are equivalent and are useful in different settings. Some additional details of the transition function are given in Appendix B.

3.5. Objective Function

We maximize the net contribution, consisting of rewards minus costs. We let *contributions* refer to parameters that can be positive (the reward of moving a train or

arriving at the shop on time, returning foreign power to the owning railroad), or negative (trains departing late, using a number of locomotives that is greater or less than the goal, arriving late to a shop appointment, holding on to foreign power). The elements of the objective function are given by the following:

c^R_{tad} = Contribution earned from acting on a locomotive $a \in \mathscr{A}$ with a decision $d \in \mathscr{D}^R$. For a decision $d \in \mathscr{D}^D$ to move a train with attribute b_d, the cost considers the desirability of using a particular type of locomotive to the type of train.

c^D_{tbd} = Contribution earned from moving a train with an attribute vector $b \in \mathscr{B}$ with a decision $d \in \mathscr{D}^D$. Higher priority trains (e.g., merchandise trains) carry higher rewards than less service sensitive freight (e.g., coal trains).

c^H_{tb} = Cost per time unit paid for holding a train $b \in \mathscr{B}$.

c^E_{tb} = Cost per power unit paid for repositioning locomotives ("empty moves") on a train $b \in \mathscr{B}$.

c^G_{tb} = Reward earned from adding one unit of power to fulfill the goal requirement of train $b \in \mathscr{B}$.

c^K = The cost of splitting a consist of locomotives into two parts.

c^F = The cost of leasing one foreign locomotive per time period.

An important dimension of the model involves the routing of power to shop locations for repair and maintenance. For now, we define the shop contribution function using the following:

$C^{ShpRte}_t(S_t, x_t)$ = Total contribution across the system for shop routing, capturing on-time performance and utility of locomotives as they are routed to shop, incurred at time t.

$C^{ShpCong}_{ts}(R^{Shp}_{ts})$ = A nonlinear function capturing the value of having R^{Shp}_{ts} locomotives at shop s at time t.

The functions $C^{ShpRte}_t(S_t, x_t)$ and $C^{ShpCong}_{ts}(R^{Shp}_{ts})$ are described in detail in §7.

The total contribution for time period t is then given by the following function:

$$\begin{aligned} C_t(S_t, x_t) = {} & \sum_{a\in\mathscr{A}} \sum_{d\in\mathscr{D}^R} c^R_{tad} x_{tad} + \sum_{d\in\mathscr{D}^D} \sum_{b\in\mathscr{B}} [c^D_{tbd} x^D_{tb} + c^H_{tb} w_{tb} \\ & + c^G_{tb} y^G_{tb} + c^E_{tb} y^E_{tb}] + \sum_{k\in\mathscr{K}_t} c^K z_{tk} + \sum_{f\in\mathscr{F}} c^F R^F_{tf} \\ & + C^{ShpRte}_t(S_t, x_t) + \sum_{s\in\mathscr{S}} C^{ShpCong}_{ts}(R^{Shp}_{ts}). \end{aligned} \tag{6}$$

We write the dependence of the costs and contributions on the state S_t to reflect the fact that the contribution function depends on data that varies over time. The variables $(w_{tb}(x_t))$, (y^G_{tb}), (y^E_{tb}), (z_{tk}), (R^{Shp}_{ts}), and (R^F_{tf}) are uniquely inferred from x_t using operational constraints that we explain in detail in Appendix C.

If our problem is deterministic, which means that $\hat{R}_t$ and $\hat{D}_t$ are zero for all time periods, we can write the objective function as

$$\max_{x_t\in\mathscr{X}_t,\, t=0,\ldots,T} \sum_{t=0}^{T} C_t(S_t, x_t) \tag{7}$$

subject to the constraints $x_t \in \mathscr{X}_t$, reflecting constraints at a point in time t, and the transition constraints (3) and (4) that link activities across time.

If we need to capture an exogenous information process, we introduce a *decision function*, or *policy*, $X^\pi_t(S_t)$, which depends on the state variable $S_t = (R_t, D_t)$. The policy is designed to return a feasible vector $x_t = X^\pi_t(S_t)$, where $x_t \in \mathscr{X}_t$. The state variable evolves according to the transition function $S_{t+1} = S^M(S_t, x_t, W_{t+1})$, where W_{t+1} represents the exogenous information received during time interval $t+1$, including a sample realization of $\hat{R}_{t+1}$ and $\hat{D}_{t+1}$. The optimization problem is now one of finding the best policy $X^\pi_t(S_t)$ in a set Π. The objective function for the stochastic problem is now written

$$\max_{\pi\in\Pi} \mathbb{E}\left\{ \sum_{t=0}^{T} C(S_t, X^\pi_t(S_t)) \right\}. \tag{8}$$

Here, the decision function $X^\pi_t(S_t)$ is designed to return a vector $x_t \in \mathscr{X}_t$, while the state variables are linked using $S_{t+1} = S^M(S_t, x_t, W_{t+1})$.

The deterministic optimization problem defined by (7) is well understood, but that does not mean the problem is easy. As we show below, there are important versions of the locomotive optimization problem that can be solved by commercial solvers. However, it is not hard to create problems that cannot be solved as a deterministic integer program.

The stochastic version, however, is an entirely different matter. Even small stochastic optimization problems can be computationally intractable. In §6, we describe an algorithmic strategy based on ADP, which we use to solve both deterministic and stochastic versions of the problem.

We begin by describing the deterministic integer programming models. We first describe the simplified scheduling model known as PLASMA/SC, which produces a rough operating plan; this model is used by Norfolk Southern in production to provide approximate, high-level scheduling instructions over an entire week. We present this model as an instance of a practical, implementable model that can be solved optimally using commercial solvers. We then describe PLASMA/MC, which uses a simplified attribute vector, producing a deterministic integer program that can be solved exactly. This model is used only for benchmarking the ADP algorithm.

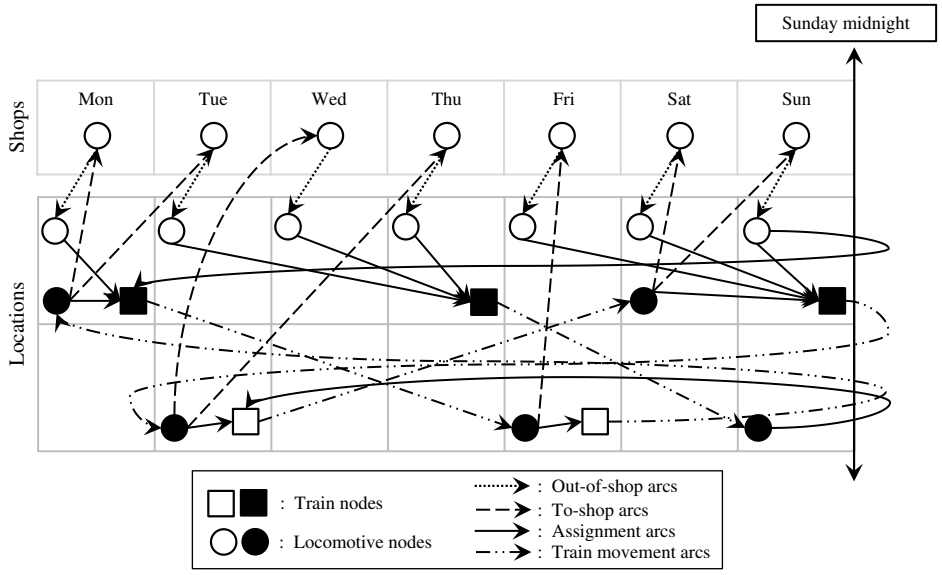

Figure 2 Network Representation of the Single Commodity Planning Model

4. The Deterministic Single Commodity Model

The single commodity planning model, PLASMA/SC (see Figure 2), is used within Norfolk Southern to provide scheduling guidelines of the following form: locomotives on inbound trains A and B should be used on outbound trains C and D. There was no expectation that this plan would be followed all of the time, but rather is only used as a guide in the field. Exceptions are made routinely as dispatchers have to take into consideration all of the issues described in the detailed model. The model is highly simplified, but it can be solved optimally using CPLEX in under two minutes for a weeklong horizon. It provides value in its ability to take a global perspective of the entire network. It is our belief that PLASMA/SC provides valuable high-level guidance to the railroad, and could be used to guide more detailed operational models such as PLASMA/MA.

The full model is described in Appendix E. The features of the model include the following:

- Starting and ending locomotive inventories are constrained to be the same, making the plan repeatable from one week to the next.
- There are bonuses encouraging flow patterns that are repeatable from one day to the next, which simplifies operations.
- The model uses a first-in, first-out assignment protocol, which is easier for yards to manage.
- The model contains simplified logic to minimize the number of times that consists have to be broken.
- It uses simplified logic for shop routing and foreign power.

It was critical that PLASMA/SC be solvable using a commercial solver such as CPLEX that ensures that flows are optimized globally over the network. For this reason, several simplifications had to be introduced. Locomotives were aggregated into a single type. Also, it does not route individual locomotives through the shop as we do in the multiattribute model, but it does approximate the flow of locomotives through the shop at an aggregate level to avoid artificially inflating the fleet. Finally, it does not recognize ownership and as a result it manages the flows to and from other railroads ("foreign power") at an aggregate level.

Similar issues were captured in the model (such as repeatability of the flows) described in Ahuja et al. (2005), which was solved using large neighborhood heuristics. We found that CPLEX 12.1 could solve our formulation of the problem without difficulty for the weekly train schedule. This model is currently being used at Norfolk Southern.

We tested the model on a typical data set with 2,082 trains scheduled over a week in 146 different locations. The network interacts with five foreign railroads through exchange locations. To obtain a solution within a reasonable run time, only one generic locomotive type is considered. The experiment has been conducted on a compute node with 96 GB of RAM and two Intel Xeon X5667 processors that supports 16 threads using CPLEX 12.1. For a relative gap of 0.07%, the model is able to produce a solution in 94 seconds.

We have been able to conclude from our work with PLASMA/SC that classical integer programming formulations can be very effective for simplified locomotive models. The most important simplifications for this model are the aggregation into a single class of locomotive, and the inability to delay trains.

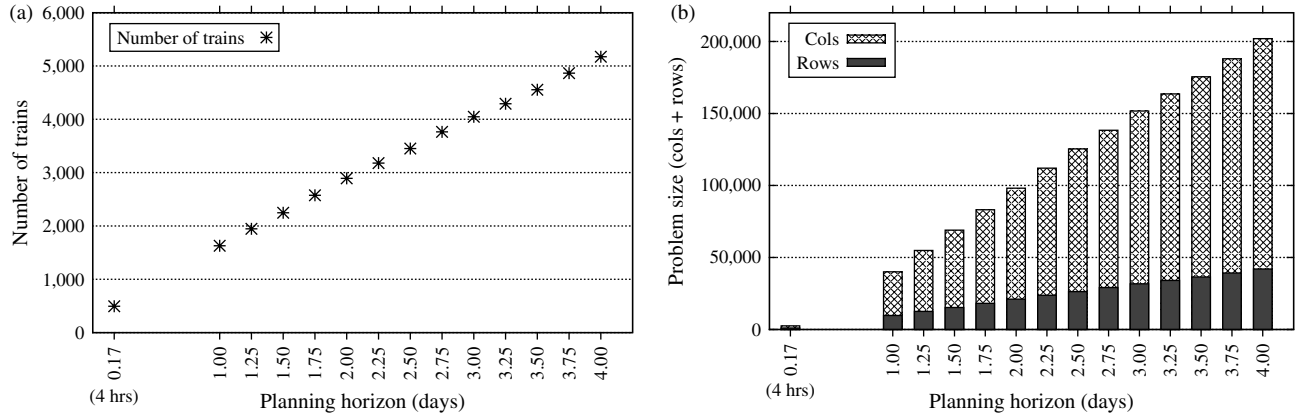

Figure 3 **Size of the Integer Programming Model as a Function of the Planning Horizon**

5. The Deterministic Multicommodity Model

PLASMA/MC is a deterministic integer programming model represented by the objective function (7), which is solved subject to the constraints $x_t \in \mathcal{X}_t$ for each time $t = 0, \ldots, T$, and the linking constraints (3) and (4). The distinguishing feature of the model is the use of a restricted set of attributes, which allows us to group locomotives into four classes (high and low horsepower, high and low adhesion), producing a classical multicommodity flow model. This is a model that can be solved by both CPLEX and (in its deterministic form) ADP, allowing us to use CPLEX as a benchmark to evaluate the quality of the ADP solution. This model also allowed us to determine the boundaries of CPLEX.

The model has been tested on a Norfolk Southern data set. The data set has 2,337 locomotives placed in 443 different yards and ready to be used at the beginning of the planning horizon. Our data set includes 9,295 trains over a seven day horizon, which proved to be beyond the capabilities of CPLEX, so we resorted to solving problems with shorter horizons. Figure 3(a) shows the number of trains as a function of the planning horizon, and Figure 3(b) shows the number of rows and columns.

This study has two purposes. First, we want to learn about the effect of the planning horizon on the solution time. This analysis allows us to understand the limits of CPLEX for solving a deterministic formulation of the model. Second, we use the results from these experiments as a benchmark for the ADP algorithm, although this comparison is reported below.

Our experiments were run with CPLEX 12.1 running on a compute node with 96 GB of RAM and two Intel Xeon X5667 processors that support 16 threads. We conducted two sets of experiments using as stopping criteria a relative gap of 0.3% in the first set and 1% in the second set. The results are shown in Figures 4(a) and 4(b).

These results show the dramatic increase in CPU times as the horizon increases, even when we relax the optimality gap. We note that caution has to be used when relaxing the optimality gap. Although a 1% gap may seem quite acceptable, it actually has been found to produce odd anomalies when detailed locomotive-to-train assignments are examined. Yet, even with a 1% gap, run times exceed five hours for a 3.5 day horizon, and a run submitted for a four day planning horizon was stopped after four days of CPU time.

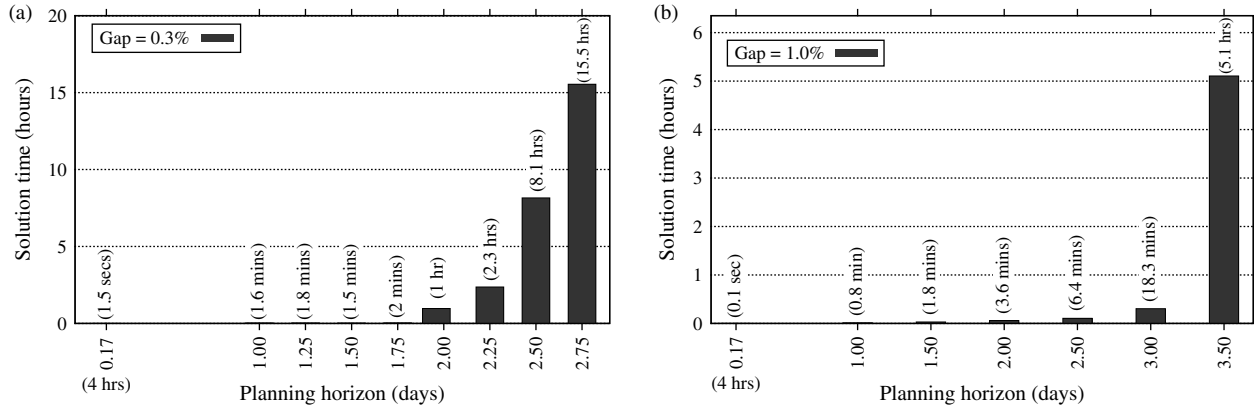

Figure 4 **CPU Time for Integer Programming Model as a Function of the Planning Horizon**

We believe that this dramatic increase in CPU times will arise in any search algorithm that attempts to solve the entire problem simultaneously over these longer horizons. The problem is that finding an improvement in an assignment at one point in time requires reoptimizing the entire path of the locomotive at all earlier and later points in time, which in turn will generally require the reoptimization of other locomotives over the entire horizon.

6. The Multiattribute Model with Approximate Dynamic Programming

The dramatic growth in run times for the aggregated version of PLASMA/MC, shown in Figure 4, demonstrates that an optimal solution of even a deterministic model over longer horizons with the full set of details is far beyond the capability of commercial solvers. It is unlikely that this is going to change even with improvements in these solvers or improvements in computer hardware. This observation motivated the development of an algorithmic strategy based on ADP.

Our original interest was a strategic planning tool that could be used for fleet sizing. Fleet sizing is a particularly difficult problem, because it requires that the model accurately capture locomotive productivity. This requires knowing that if a train needs an LSL locomotive (which limits speed), we cannot move the train unless one of the locomotives in the consist has this characteristic. We also have to manage routing locomotives to shop locations, while simultaneously choosing locations to balance shop congestion. In addition, sometimes we will move a train with three locomotives, even if it only needs two, in order to save the cost of breaking a consist. Alternatively, we may need to pull additional locomotives to balance locomotive inventories around the network. Finally, we also have to route locomotives owned by other railroads ("foreign power") back to the owning railroad.

One algorithmic strategy is to use heuristics such as the metaheuristics suggested in Glover and Kochenberger (2003) or large neighborhood search algorithms proposed in Ahuja et al. (2005). We attempted the use of heuristics to solve the local locomotive assignment problem in a yard and found that it worked quite well most of the time. However, from time to time it would provide answers that were clearly incorrect, causing considerable problems in identifying the cause of a problem (data?, model?, algorithm?). Furthermore, heuristics appeared to be an interesting strategy for deterministic problems, but did not provide a natural path to handle uncertainties in an elegant way. For this reason, we adopted the strategy of ADP, which can be used for both deterministic and stochastic problems.

Our goal with ADP is to show the following: (1) ADP allows us to capture a high level of detail, producing a model that calibrates accurately against historical performance using a rigorous modeling and algorithmic framework. (2) We wish to evaluate the quality of the ADP solution against the optimal solution of the largest model we can solve using integer programming on a deterministic model. (3) Finally, we wish to show that an ADP policy that was trained using stochastic travel times produces a solution that is more robust than an ADP policy trained on deterministic travel times (which is quite close to the optimal solution obtained using integer programming).

6.1. The ADP Algorithm

We develop the ADP strategy in the context of a stochastic model, and then show how it can be trivially adapted to a deterministic model. We start by writing Bellman's equation as

$$V_t(S_t) = \max_{x_t \in \mathscr{X}_t} (C(S_t, x_t) + \mathbb{E}\{V_{t+1}(S_{t+1}) \mid S_t\}).$$

To avoid the expectation, we use the pre-decision and post-decision resource states (see Equation (5)) to break the Bellman's equation into two equations:

$$V_t(S_t) = \max_{x_t \in \mathscr{X}_t} (C(S_t, x_t) + V_t^x(S_t^x)), \tag{9}$$

$$V_t^x(S_t^x) = \mathbb{E}\{V_{t+1}(S_{t+1}) \mid S_t^x\}. \tag{10}$$

Since we will never know $V_t^x(S_t^x)$ exactly, we replace it with an approximation $\bar{V}_t(S_t^x)$.

Also, since x_t is a fairly high-dimensional integer vector, we would like to draw on the power of math programming algorithms, which limits the types of approximations we can use. Early in the project, we used approximations that were linear in the resource variable R_t. Recognizing that $S_t^x = (R_t^x, D_t^x)$, these linear approximations had the form

$$\bar{V}_t(S_t^x) = \sum_{a \in \mathscr{A}^1} \bar{V}_{ta} R_{ta}^x,$$

where R_{ta}^x can be written

$$R_{ta}^x = \sum_{a' \in \mathscr{A}^1} \sum_{d \in \mathscr{D}^R} \delta_{t,a',ad} x_{ta'd}.$$

Note that we are approximating locomotives at the aggregation level represented by $\mathscr{A}^1$, which captures location, special equipment, the type of locomotive and if it is attached to a train. Note that our approximation ignores the value of trains that are being delayed. This approximation appears to work well enough, probably because delayed trains are not that common.

We worked with linear approximations for several years. Although they initially produced good results

(and are much simpler to work with), we ultimately found them to be unstable. We then made the transition to separable, piecewise linear value function approximations following the work of Godfrey and Powell (2001, 2002) and Topaloglu and Powell (2006). These had the general form

$$\bar{V}_t(S_t^x) = \sum_{a \in \mathcal{A}^1} \bar{V}_{ta}(R_{ta}^x).$$

Piecewise linear functions are much harder to work with, primarily because decisions have to be made over time, and travel times range from a few hours to a day or more. If you wish to move additional locomotives into a region Wednesday afternoon, you can satisfy this need by moving locomotives over long distances starting early on Tuesday, or over shorter distances on trains from nearby locations that depart Wednesday morning. Very specialized algorithms are required to handle the problem of multiperiod travel times using nonlinear value function approximations (see Godfrey and Powell 2002 and Topaloglu and Powell 2006 for a full description of how this problem is solved).

Over time, however, we found that even the nonlinear approximations did not work well enough to satisfy the careful analysis of the dispatchers within Norfolk Southern. The problem could be traced to the use of separable approximations. We might get approximately the right number of locomotives of each type, but in aggregate we found that we simply had too many locomotives in some locations.

To overcome this problem, we introduced (for the first time in our research) two layers of value function approximations that work at two different levels of aggregation. Let $\mathcal{A}^g$ be the attribute space of locomotives at aggregation level $g \in \mathcal{G} = \{1, 2\}$, where $\mathcal{A}^1$ is the more disaggregate level of aggregation (but more aggregate than $\mathcal{A} = \mathcal{A}^0$), which captures locomotive type, special equipment, and location, while $\mathcal{A}^2$ ignores the locomotive type (but still captures location).

We create two sets of value function approximations that we represent using $\bar{V}_t^g(R_t^g)$, where R_t^g is the resource vector using attribute space $\mathcal{A}^g$. We then created a new value function approximation given by a weighted sum of $\bar{V}_t^g(R_t^g)$ for $g \in \mathcal{G}$, given by

$$\bar{V}_t(R_t) = \sum_{g \in \mathcal{G}} \alpha^g \bar{V}_t^g(R_t^g), \tag{11}$$

$$\bar{V}_t^g(R_t^g) = \sum_{a \in \mathcal{A}^g} \bar{V}_{ta}^g(R_{ta}^g), \quad g \in \{1, 2\}, \tag{12}$$

$$\sum_{g \in \mathcal{G}} \alpha^g = 1. \tag{13}$$

Our locomotive assignment policy is now given by

$$X_t^\pi(S_t) = \arg\max_{x_t \in \mathcal{X}_t} (C(S_t, x_t) + \bar{V}_t(R_t^x)). \tag{14}$$

Figure 5 illustrates the locomotive assignment problem, showing individual locomotives to the left being assigned to individual trains. The assignment problem includes all locomotives and trains that are available at time t, or will become available within a four hour window (the time interval from t to $t+1$). The value of locomotives in the future are captured at two levels of aggregation. The detailed algorithm, without shop routing, is given in Figure 6.

The algorithm progresses by simulating policy $X_t^\pi(S_t)$ over the entire simulation horizon using $\bar{V}_t(R_t)$. Computing $X_t^\pi(S_t)$ requires solving an integer program subject to constraints $x_t \in \mathcal{X}_t$ on the availability of locomotives captured by the high-dimensional vector R_t.

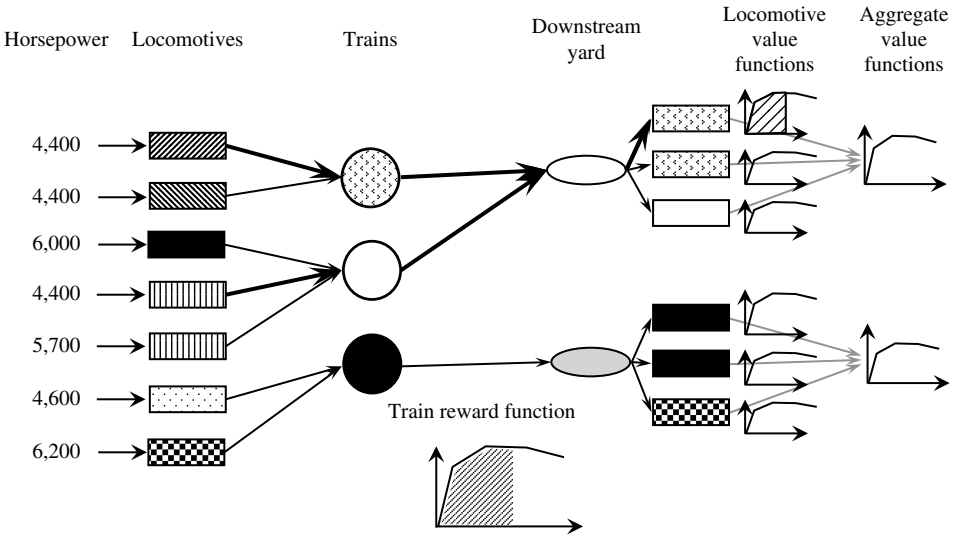

Figure 5 Illustration of Locomotive Assignment Policy Using Hierarchical Value Function Approximations

Step 0. Initialization:

Step 0a. Initialize an approximation for the value function $\bar{V}_t^{g,0}(R_t^x)$ to zero for all post-decision states R_t^x, time periods $t = \{0, 1, \ldots, T\}$, and aggregation levels $g = \{1, 2\}$.

Step 0b. Set $n = 1$.

Step 0c. Initialize S_0^1. If the model is being run as a strategic planning system, all locomotives start in a common "node in the sky" location; if the model is being run as an operational planning system, all locomotives are initialized at a given starting position.

Step 1. Choose a sample path ω^n. If the model is being run in a deterministic model, the sample path is always the same.

Step 2. Do for $t = 0, 1, \ldots, T$:

Step 2a: Choose a locomotive assignment using

$$X_t^\pi(S_t) = \arg\max_{x_t \in \mathscr{X}_t} \left(C(S_t^n, x_t) + \gamma \sum_{g \in \mathscr{G}} \alpha^g \sum_{a \in \mathscr{A}^g} \bar{V}_{ta}^{g,n-1}(R_{ta}^{x,n}) \right),$$

where $R_t^{x,n}$ is the post-decision resource vector given by (5). Let x_t^n be the vector of assignments of locomotives to trains, including decisions to hold locomotives and trains. Compute $\hat{v}_{ta}^n$ giving the marginal value of additional locomotives of type $a \in \mathscr{A}^g$ for $g \in \mathscr{G}$.

Step 2b. Update $\bar{V}_{t-1}^{g,n-1}$ with the slopes $\hat{v}_t^{g,n}$ using the CAVE algorithm.

Step 2c. Sample $W_{t+1}^n = W_{t+1}(\omega^n)$ and compute the next state $S_{t+1}^n = S^M(S_t^n, x_t^n, W_{t+1}^n)$.

Step 3. Calculate the set of paths $\mathscr{P}^{Shp} = (\mathscr{P}_{tis}^{Shp})$ featuring different travel times for getting power to shops using the train departures given by $x^n = (x_t^n)$ as described in §7.

Step 4. Increment n. If $n \le N$ go to Step 1.

Step 5. Return the value functions $(\bar{V}_t^{g,N})_{t=1}^T$.

Figure 6 ADP Algorithm for the Locomotive Assignment Problem

To compute the marginal value of an additional locomotive of type $a \in \mathscr{A}^g$, we first define

$$F(R_t) = \max_{x_t \in \mathscr{X}_t(R_t)} (C(S_t, x_t) + \bar{V}_t(S_t^x)),$$

where R_t is imbedded in the constraint set $\mathscr{X}_t(R_t)$ (where we have written the dependence explicitly here). We approximate the marginal value of an additional locomotive with attribute a by computing the dual variables $\hat{v}_{ta}^{g,n}$ of the linear relaxation. We tried computing numerical derivatives of the form

$$\hat{v}_{ta}^{g,n} = F(R_t^g + e_{ta}) - F(R_t^g),$$

where e_{ta} is a vector of zeros with an element for each $a' \in \mathscr{A}^g$ and a one for element a. Numerical derivatives did not work because adding one additional locomotive would not have any effect if two locomotives were needed to move the train. We computed the duals for all $a \in \mathscr{A}^g$.

The value functions are built using an iterative adaptive learning algorithm where, at iteration n, the information (in the form of dual variables $\hat{v}^{g,n}$) needed to update these functions is gathered and stored as the algorithm steps over time. Then, this information is used at the end of the iteration to update the functions to be used in the next iteration as follows

$$\bar{V}_{t-1}^{g,n} = U(\bar{V}_{t-1}^{g,(n-1)}, \bar{S}_{t-1}^{g,n}, \hat{v}_t^{g,n}), \quad g = 1, 2. \tag{15}$$

In the equation above, $U(\cdot)$ is a generic updating function, $\bar{V}_{t-1}^{g,n} = (\bar{V}_{t-1,a}^{g,n})_{a \in \mathscr{A}^g}$, and $\hat{v}_t^{g,n} = (\hat{v}_{ta}^{g,n})_{a \in \mathscr{A}^g}$ is the vector of derivatives at time period $t-1$ at iteration n. We use the CAVE algorithm first given in Godfrey and Powell (2001) (see also Powell 2011, Chapter 13). The CAVE algorithm is specifically designed to maintain concavity, and has been shown to produce high-quality solutions in the context of fleet management problems in truckload trucking and the management of freight cars for railroads.

The real value of solving locomotive assignment problems of the form illustrated in Figure 5 is speed. Whereas it may take five hours to solve a problem over a 3.5 day horizon (using a generous 1% optimality gap), solving a single locomotive assignment problem such as that shown in Figure 5 takes around three seconds using CPLEX, despite being a problem with thousands of integer variables. Shallow problems appear to be extremely easy to solve. Even more important is that the problem in Figure 5 captures each locomotive with the full set of attributes, including modeling the time of availability down to the minute.

6.2. Benchmarking Against Optimal Deterministic Solution

As an initial test of our ADP algorithm, we solved a simplified version of the deterministic locomotive planning problem (no train delay, no shop routing, no foreign exchange, three days horizon) as a single integer program. Then, we solved the same problem with the ADP algorithm proposed in this section. Figure 7 shows the ADP objective function as a percent of the optimal from CPLEX. After 90 iterations, ADP produced a solution within half of 1% of the solution produced by CPLEX on a data set with a 3.5 day horizon (the longest that we could solve using CPLEX). Later, we present a more comprehensive set of tests.

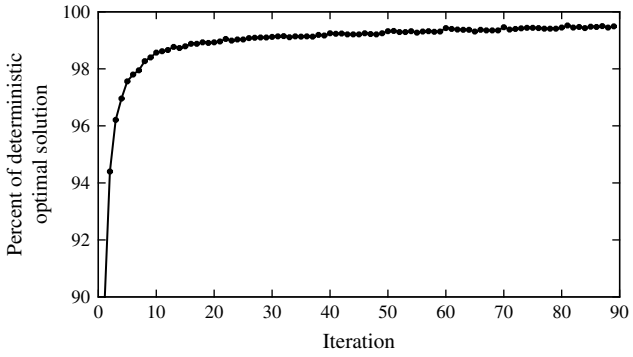

Figure 7 The ADP Objective Function as a Percent of the Optimal Deterministic Solution Obtained Using CPLEX, for a Problem with a 3.5 Day Planning Horizon

7. Shop Routing

Shop routing is one of the most difficult operational challenges for the freight railroads in the United States. We have to get a locomotive due at the shop at time τ as close to this time without arriving late. The locomotive has to move primarily on scheduled trains (which are subject to random delays). In addition, we would like the locomotive to be as productive as possible while moving toward its shop. As a final twist, we have some flexibility in the choice of shop we choose to send a locomotive, and the system has to balance congestion across the shops.

We first define the following:

$\mathcal{S}$ = Set of shop locations.

$\mathcal{P}_{tis}^{Shp}$ = A set of paths featuring different travel times for getting power from yard i at time t to shop s over scheduled trains.

$\tau_{tis}^{Shp}(p)$ = The time required to traverse path $p \in \mathcal{P}_{tis}^{Shp}$.

Using this notation, we define three sets of costs.

$c_{ts}^{ShpTme}(p)$ = Bonus/penalty from arriving early/late for getting a locomotive to shop s at time τ when using path $p \in \mathcal{P}_{tis}^{Shp}$, where the function is given in Figure 8(a).

$c_{ts}^{ShpUtil}(p)$ = Cost attributed to the utilization of a locomotive getting to shop when using path $p \in \mathcal{P}_{tis}^{Shp}$ for getting a locomotive to shop s. This term captures the costs for setouts that require breaking a locomotive from a consist.

$C_{ts}^{ShpCong}(R_{ts}^{Shp})$ = A piecewise linear and concave congestion function for the maintenance and repair services at shop $s \in \mathcal{S}$ when there are R_{ts}^{Shp} locomotives in shop s at time t, where the function is given in Figure 8(b), which depicts an ideal level of shop utilization, with lower rewards when there are too many or too few locomotives.

The shop routing model considers the marginal contributions $c_{ts}^{ShpTme}(p)$ and $c_{ts}^{ShpUtil}(p)$ per locomotive, and the total contribution $C_{ts}^{ShpCong}(R_{ts}^{Shp})$, which reflects the congestion when there are R_{ts}^{Shp} locomotives in shop s at time t.

We solve the shop routing problem using an imbedded dynamic program that produces a set of paths that can get a locomotive from a particular yard i at a point in time t to a shop s by moving on a sequence of trains. We do this by creating a graph of movements resulting from a forward pass of the algorithm, illustrated in Figure 9. We then solve a shortest path problem (illustrated in Figure 10) to get us from a yard i at time t to shop s at a time $t' > t$, where t' is set to $t+24, t+48, \ldots,$ yielding a set of shortest paths that get to the shop at different points in time in the future. These paths make up the set $\mathcal{P}_{tis}^{Shp}$, with associated travel times $\tau_{tis}^{Shp}(p)$ for $p \in \mathcal{P}_{tis}^{Shp}$.

When we are considering an assignment of a locomotive with attribute a to train $d \in \mathcal{D}^B$, it implies an arrival to a yard j at a time t' corresponding to the estimated time of arrival of train b_d. If the locomotive needs to get to a shop, we search over all paths $p \in \mathcal{P}_{t'js}^{Shp}$ for all shops $s \in \mathcal{S}$ and we pick the one with the largest total value of time and utility contributions. That is, we compute

$$c_{tb}^{Shp} = \text{Highest contribution of on-time plus utility across path shops and all paths to each shop,}$$
$$= \max_{p \in \mathcal{P}_{t'js}^{Shp},\, s \in \mathcal{S}} \{c_{ts}^{ShpTme}(p) + c_{ts}^{ShpUtil}(p)\}.$$

The total shop cost for a vector x_t is then given by

$$C_t^{ShpRte}(S_t, x_t) = \sum_{a \in \mathcal{A},\, d \in \mathcal{D}^B} c_{tb_d}^{Shp} x_{tad}.$$

This shop cost is included in the contribution function $C_t(S_t, x_t)$ in Equation (6), forcing the model to balance

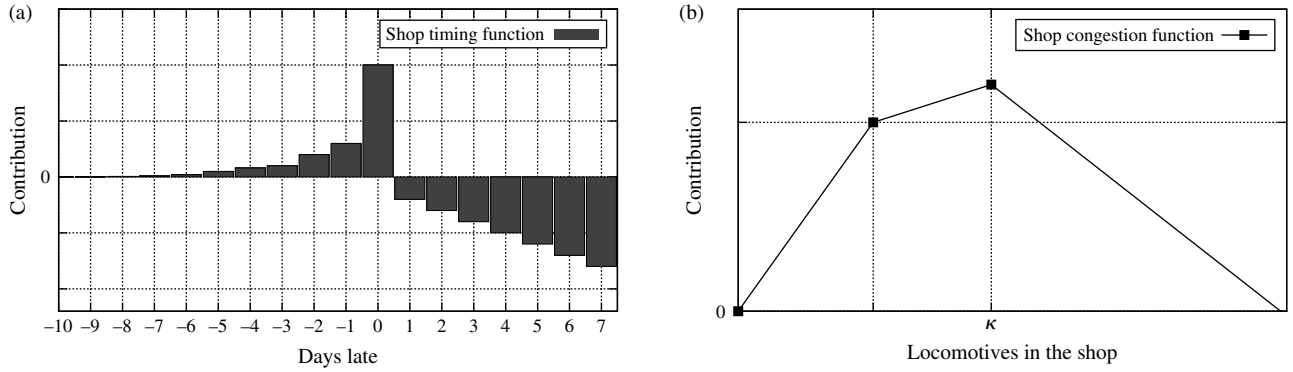

Figure 8 **(a) Contribution for Arriving to Shop as a Function of the Arrival Time Relative to the Scheduled Arrival Time, and (b) Shop Congestion Function, Showing Increasing Value Up to the Capacity of the Shop, Followed by a Downward Sloping Function If the Number of Locomotives Exceeds Capacity, Implying Queueing**

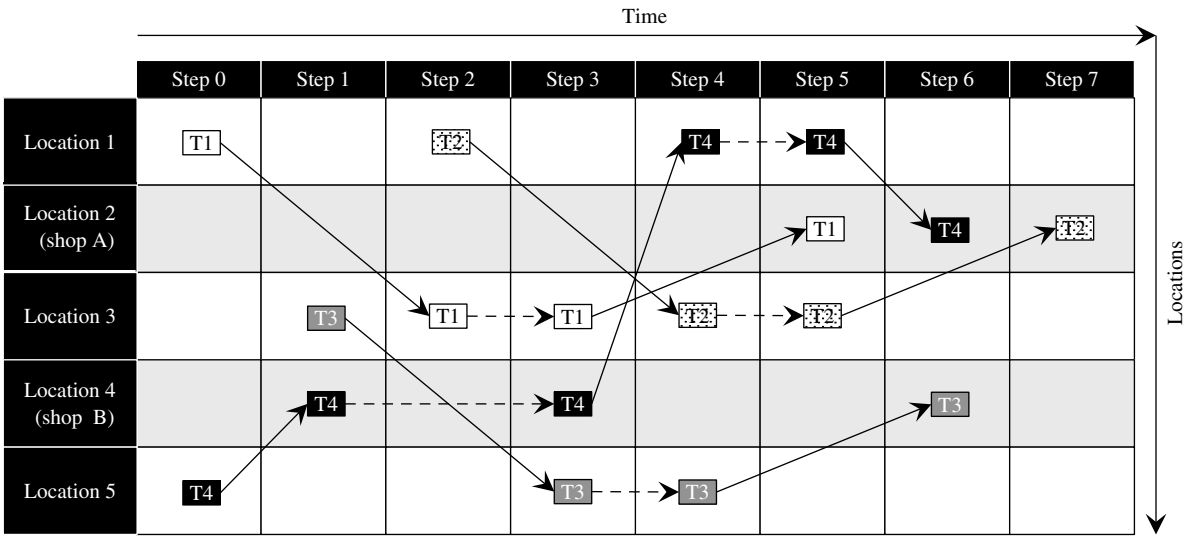

Figure 9 **Train Movements in the Current Iteration**

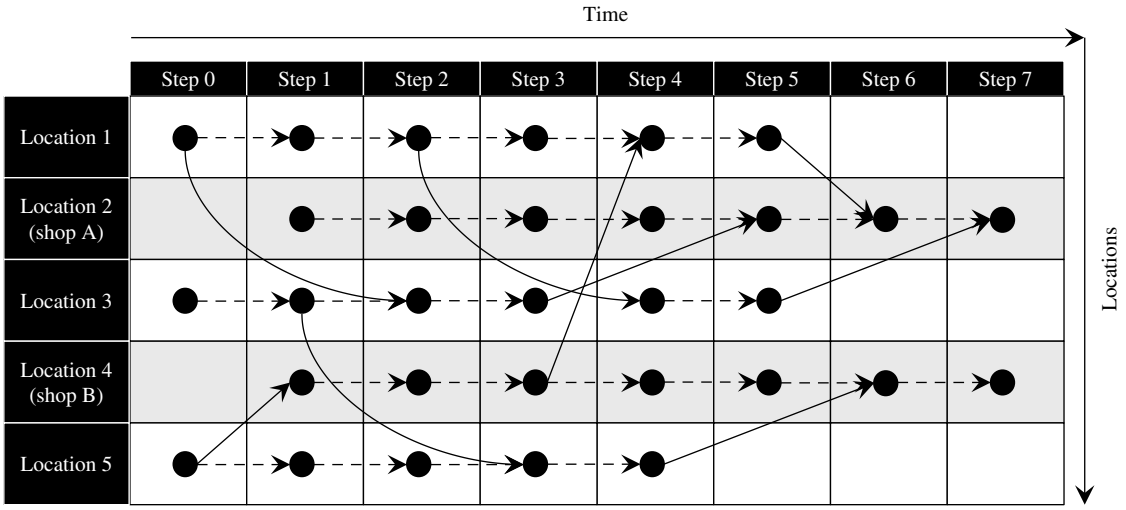

Figure 10 **Shop Routing Paths Built Using the Train Movements**

shop routing issues with the other dimensions of the locomotive management problem. We note that our shop logic depends on the results of each simulation that determines when trains actually move, and the total number of locomotives waiting in line at each location at each point in time. We learn from these activities to estimate the marginal value of assigning a locomotive to arrive at a shop at a point in time, and when constructing the train schedule for the purpose of solving the shortest path problem.

8. Model Applications

As of this writing, the multiattribute model can be run in two modes: strategic planning to determine fleet size and mix, and tactical planning to forecast/optimize activities over a one–seven day horizon. The model can also be adapted for real-time planning purposes, but this third version has not been implemented as of this time. The only difference between the two models is the initial inventory of locomotives.

The strategic model works as follows. At the beginning of the simulation horizon, all locomotives are placed in a super source, with one node for each type of locomotive. The first decision problem, $X_0^{\pi}(S_0)$ has to move locomotives from the supersource to a starting location. This is guided by the piecewise linear value functions $\bar{V}_0^{n-1}(R_0^x)$. Other than this initial step, which is distinguished only by the initial positions of locomotives in a supersource, the algorithm is no different than when the starting attribute of each locomotive is specified in a snapshot file (as is done in the tactical planning model).

The tactical planning model, on the other hand, uses an initial snapshot of all of the locomotives, which specifies the full set of attributes of each locomotive including starting location and time, as well as its maintenance status.

8.1. Calibration

Several years of calibration were devoted to the tuning of the model, algorithm, and data before the model was deemed to be calibrated. At Norfolk Southern, determining the right fleet size and mix requires trading off the expense of a larger fleet with the train delays that result from having too few locomotives. The ADP model will delay trains if there are not enough locomotives, but good estimates of train delays require that the model accurately represents operations at a surprisingly high level of detail. Our biggest challenge was the accuracy of the locomotive snapshot, where errors in the starting location would produce delays because the locomotives were not in their real position.

Considerable care was invested in a detailed analysis of individual locomotive to train assignments. Although high-level performance measures are useful, they were not enough for user acceptance. The calibration process was assisted by a powerful tool, Pilotview, developed in CASTLE Lab for analysis of the results of transportation models. See http://www.castlelab.princeton.edu/plasma.html to view a screen snapshot of Pilotview.

The calibration process involved generating train-delay curves, where we would run the model for a range of different fleet sizes, computing aggregate train delay for each fleet size. We would then compare historical train delay given the current fleet size. Figure 11(a) shows the results of the initial calibration exercise, using a data set from October 2007. The superb fit was achieved largely because of careful attention to data, refinement of the model, and improvements to the algorithm (in initial runs, the fit was terrible). Also, although the model has very few tunable parameters, there was no explicit step to fit the model using these parameters, largely because the initial problems could not have been fixed just by tuning a parameter.

Calibrating a model so that it produces the right level of train delay is actually quite difficult, and we would argue that it represents a major plateau in locomotive modeling. It requires we strike the right balance for breaking consists, accurate modeling of yard delays, and the proper amount of repositioning locomotives to balance the needs between yards. It is also necessary to model the assignment of locomotives to shops for maintenance, as well as the proper handling of foreign power. It is standard industry practice to optimize flows assuming all trains leave on time (as we do in PLASMA/SC), but such models cannot be used for fleet sizing.

After achieving the close fit in Figure 11(a), we repeated the exercise on other data sets without duplicating the calibration process. The fit shown in Figure 11(b) for a data set from March 2008, is typical of the results we would obtain. After repeating this process a number of times, Norfolk Southern gained confidence that the model was capturing the behavior of their network, and the model became accepted as their fleet planning tool.

8.2. Sensitivity and Stability Analysis

An important question concerning the model is its sensitivity and stability with respect to different parameters that influence the behavior of the model. We analyzed these questions for two parameters of considerable importance to any railroad: train speed and consist breakup cost. To address train speed, we generated train delay curves (delay as a function of fleet size) while increasing or decreasing train speeds 5%, 10%, and 15% relative to the base speed. The results are shown in Figure 12(a), which indicates that the model responds to train speeds as we would expect.

We then performed a series of runs using a fixed fleet size, where we varied the consist breakup cost. These results show that increasing the consist breakup cost produced a steady, stable reduction in the number of consist breakups. These analyses suggest that ADP produces results that respond in a stable, predictable way.

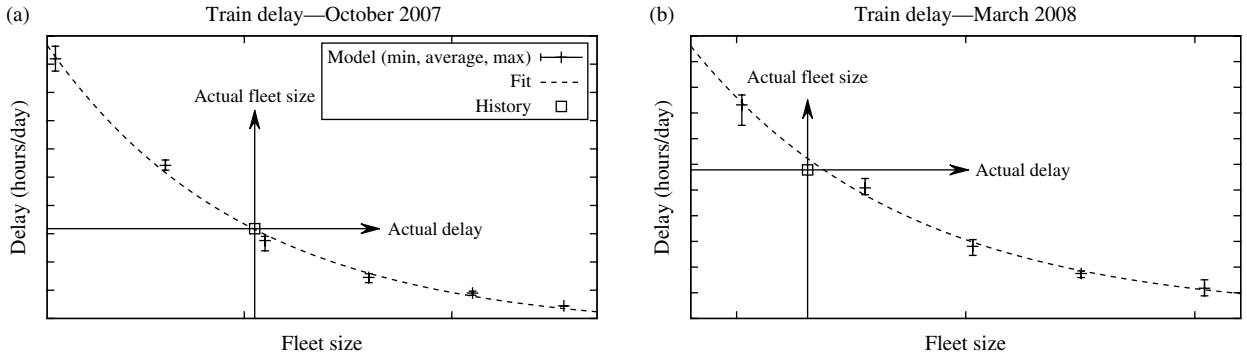

Figure 11 **Train Delay as a Function of Fleet Size, Compared to Historical Delay for October 2007 (Used for Calibration) and March 2008**

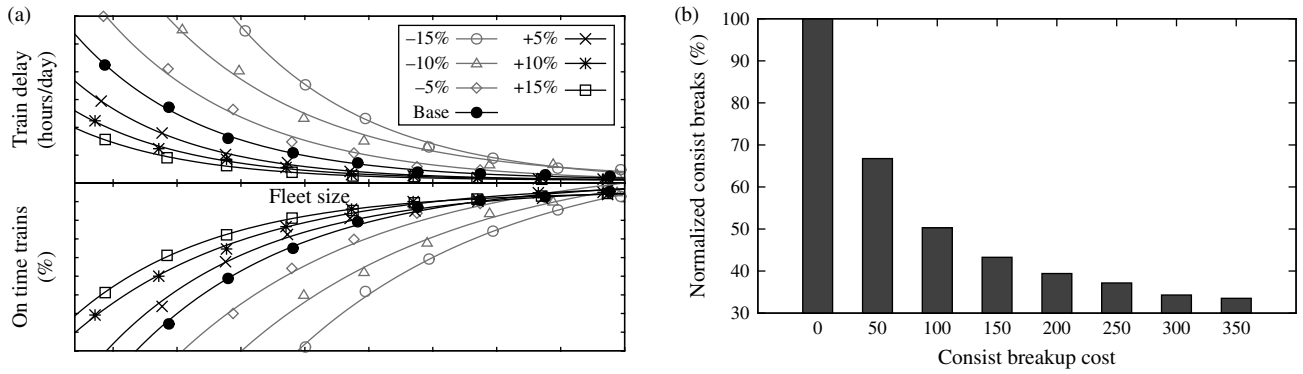

Figure 12 **(a) On-Time Trains and Train Delay, as a Function of Fleet Size, for a Range of Average Train Speeds. (b) Consist Breakups for Different Consist Breakup Costs, Normalized as a Percent of Consist Breakups When the Cost Is Zero.**

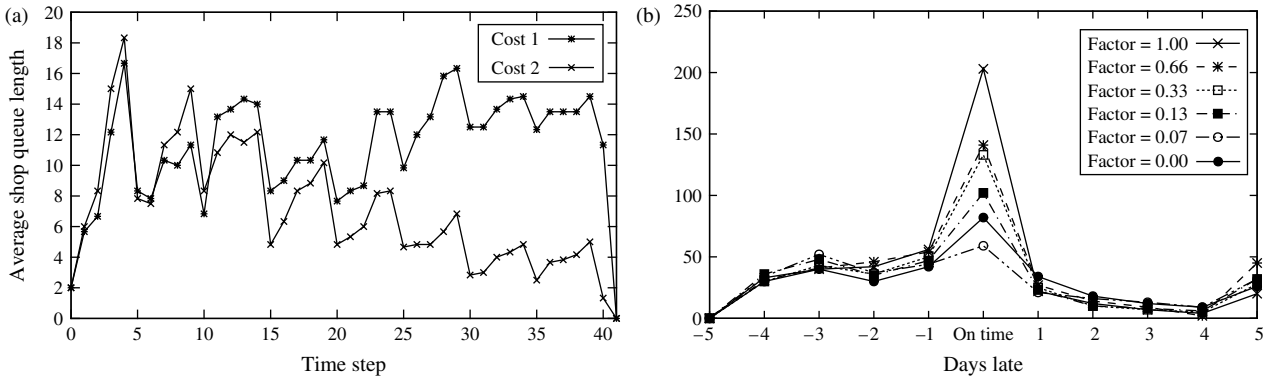

Figure 13 **Effect of the Congestion Cost on the Shop Queue (a), and the Effect of the Penalty Cost on Arrival Times to Shop (b)**

8.3. Shop Routing

We tested the ability of the model to manage congestion across shops, while still maintaining good performance in terms of getting power to a shop location on time. Figure 13(a) shows the aggregate number of locomotives sitting at different shop locations. Note that there is little the model can do in the early time periods. When a low cost is used for shop congestion, we see the number of locomotives sitting at different shops remains high throughout the simulation. When we increase the shop congestion penalty, there is a steady reduction in the number of locomotives sitting at shop locations. This suggests that we should be able to achieve higher locomotive utilization by minimizing the number of locomotives waiting in line for their shop appointment.

Figure 13(b) then shows the effect of varying the bonuses for arriving early (or penalties for arriving late) given in Figure 8. As more weight is given to the arrival time bonuses and penalties, we see a clear effect on the ability of the model to get locomotives to the shop on time. We note that even with a bonus of zero, some locomotives arrive at the shop in time because they are already at the correct location in the data set.

8.4. Testing on Stochastic Data

An advantage of ADP is the ease with which we can make the transition from solving deterministic models to stochastic models. The only difference occurs when we choose a sample path in Step 1 of the algorithm in Figure 6. If we are solving a deterministic model, the new information about locomotives and trains is always the same. When we move to solving a stochastic model, we introduce the dimension of sampling from distributions for transit times, yard delays, equipment failures, and schedule changes.

Incorporating uncertainty into the simulation of a policy is straightforward. The challenge is producing a policy that is *robust*, which is to say that it makes adaptations to deal with uncertainty more effectively. In our setting, this occurs in a natural and intuitive way. Figure 14 shows actual value functions for a particular yard at a particular point in time on the Norfolk Southern network, trained using deterministic and stochastic data. The deterministic value function rises and then quickly levels off when it reaches the number of locomotives needed for the yard at that point in time.

When we use stochastic training, the value function keeps rising. Instead of needing approximately

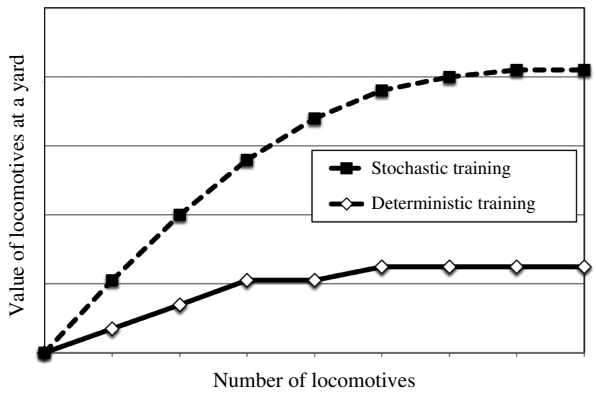

Figure 14 **Deterministically and Stochastically Trained Value Function Approximations, for a Particular Yard at a Point in Time in the Norfolk Southern Network**

three locomotives, the value function using stochastic training suggests that we need five or six locomotives, and we still derive some value from as many as seven locomotives. Interestingly, the marginal value of the first locomotive is much higher when using stochastic training than deterministic training. We believe this arises because locomotives are simply more valuable when the network is stressed by the uncertainty railroads experience from variations in travel times. The value of a locomotive is magnified throughout the network, because of the potential value added at future points in time, at future locations.

We next ran two sets of simulations. In the first, we trained the value functions on deterministic data, and then ran a series of simulations where we simulated variations in transit times and yard delays. In the second, we trained the value functions on stochastic data, and then performed the same set of simulations with variations in transit times and yard delays. Figure 15(a) shows the resulting inventories, demonstrating that stochastically trained value functions lead to behaviors where yards are more likely to hold onto power when possible.

Figure 15(b) then shows the resulting train delays when we simulate the deterministically and stochastically trained value functions. The delays with stochastically trained value functions are not just lower, but are considerably less volatile.

The stochastically trained value function approximations produce the intuitive behavior of responding to uncertainty by maintaining buffer stocks of locomotives. This logic would be hard to mimic in a rule-based simulator in part because the buffer stocks differ not just from one yard to the next, but also by day of week. This behavior reflects the well-known phenomenon that the utilization of locomotives varies by day of week. Value function approximations also make it possible for the model to make trade-offs between holding power in one yard versus repositioning it to another yard where it offers higher value.

9. Concluding Remarks

This paper has explored three classes of optimization models for locomotive planning: a streamlined single commodity model, a more detailed multicommodity model, and a multiattribute model. We have then considered two algorithmic strategies: integer programming (using CPLEX) and ADP. We showed that CPLEX could easily handle the streamlined single commodity problem, which is a model that is used in production at Norfolk Southern to provide high-level schedule guidance. However, CPLEX struggled with horizons more than three days for the more detailed model.

ADP, which can be described as an "optimizing simulator," uses feedback learning through value function approximations to decompose large problems into a sequence of smaller ones. One of the unexpected outcomes of this transition was the surprising reduction in CPU times, even when CPLEX was used to optimize locomotives over the entire railroad. ADP grows linearly with the number of time periods, implying that we could run simulations over a month or more without difficulty. The tactical model can be run in

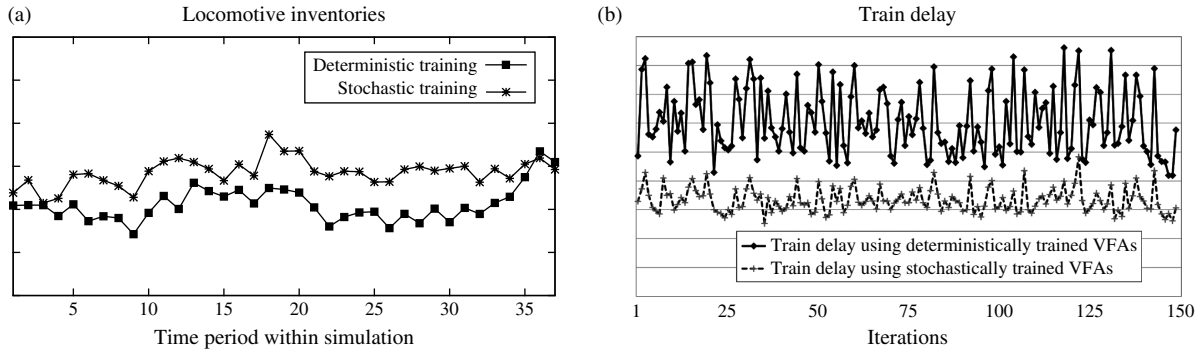

Figure 15 **(a) Systemwide Locomotive Inventories Using Value Functions Trained on Stochastic and Deterministic Data, and (b) Total Train Delay When Using Deterministically and Stochastically Trained Value Function Approximations**

a production setting with a one-week horizon, with updates every two minutes.

The research showed that an ADP-based model could handle a wide range of issues, spanning the goals and constraints that guide consist formation, management of foreign power, and the complex problem of simultaneously routing power to shops while managing congestion delays across the shops. We also found that it could produce robust policies that handle uncertainty in transit times and yard delays using simple, intuitive logic.

Acknowledgments
This research into the approximate dynamic programming algorithms was supported in part by the Air Force Office of Scientific Research (AFOSR) [Grant contract FA9550-11-1-0172]. The adaptation to the locomotive planning problem was funded by the Norfolk Southern Corporation.

Appendix A. The Attribute Vectors
Each vector of locomotive attributes $a \in \mathcal{A}$ contains the following 14 elements:

a_1: decision time, which gives the discrete time t that determines when we make a decision about assigning the locomotive;
a_2: current or inbound location;
a_3: locomotive ID;
a_4: owner;
a_5: type;
a_6: axles per unit;
a_7: horsepower per unit;
a_8: actionable (or available) time (also called the estimated time of arrival);
a_9: current consist;
a_{10}: current train;
a_{11}: maintenance date;
a_{12}: shop repair status;
a_{13}: previous train symbol;
a_{14}: special equipment type installed on the locomotive (LSL, CSS, FTS).

The attributes $a_3, \ldots, a_7, a_{14}$ are static. The remaining attributes are dynamic.

The attribute vector b for trains is given by the following:

b_1: decision time (normally the departure time, rounded down to the nearest four hour interval);
b_2: origin;
b_3: destination;
b_4: train ID;
b_5: type (merchandise, coal, grain, unit);
b_6: segment sequence;
b_7: arrival time to current location;
b_8: dwell duration at current location;
b_9: scheduled departure time;
b_{10}: duration;
b_{11}: expiration time (the train is canceled if held beyond this time);
b_{12}: minimum power;
b_{13}: goal power;
b_{14}: maximum power;
b_{15}: special locomotive equipment type required;
b_{16}: number of special locomotives required;
b_{17}: tonnage;
b_{18}: miles;
b_{19}: revenue;
b_{20}: previous train;
b_{21}: next train;
b_{22}: accumulated delay;
b_{23}: train symbol.

All of the dynamic attributes of a train are modified when acted on with an assignment decision. In the case of a decision to hold a train, the only attribute that changes is the decision time b_1, since we move the train to the next decision subproblem. Furthermore, the arrival time to current location attribute b_7 is determined by the departure time of the previous train, if b_7 exists. Trains can be delayed up to an expiration time and cannot depart before the scheduled departure time.

Appendix B. The Transition Function
According to our modeling framework, when a locomotive with attribute $a \in \mathcal{A}$ is acted on at time period t with a decision $d \in \mathcal{D}$, the locomotive will be available in a future time period $t' > t$ with an attribute $a' = a^M(a, d, W)$ (Equation (1)). Similarly, when a train $b \in \mathcal{B}$ is moved with a decision $d \in \mathcal{D}^D$, the train will be available in the future with a modified attribute $b' = b^M(a, d, W)$ (Equation (2)). In this appendix, we describe how the arguments of a^M and b^M are used to obtain a' and b'.

It is important to model the amount of time required to complete a decision. For this we define

τ_{td} = the expected duration of a decision $d \in \mathcal{D}$ made at time t;
$\tau_{t+1,d}$ = a random variable at time t that gives the duration of decision d based on information available at time $t+1$.

These durations can be transit times, yard delays, set-off times (when breaking consists), and the time required to finish a shop appointment. If a train departs at time $t = 08{:}00$ on a move that should take $\tau_{td} = 10$ hours (based on what we know at time t), we may get updates at each decision epoch. Thus, by time $t+1$, corresponding to time 12:00, we may have learned that the transit time has increased to $\tau_{t+1,d} = 11.2$ hours.

The process of updating the attribute vector a_t depends on the nature of the decision. For a hold decision, the only change is the decision time. When a locomotive moves a train, its location changes, along with its available time (estimated time of arrival) and possibly its maintenance status. This transition might be written as

$$a_{t+1,i} = \begin{cases} \max\{b_{t1}, b_{t9}\} + w_{tb} + \tau_{t+1,d}, & \text{if } i = 1, 8 \\ b_{t3}, & \text{if } i = 2 \\ b_{t4}, & \text{if } i = 9 \\ b_{t,10}, & \text{if } i = 8 \\ b_{t,21}, & \text{if } i = 10 \\ a_{t+1,12}, & \text{if } i = 12 \\ b_{23}, & \text{if } i = 13 \\ a_i, & \text{otherwise.} \end{cases}$$

If we are solving a stochastic model, we would have to randomly sample the transit time $\tau_{t+1,a}$, and the repair status $a_{t+1,12}$. The point is to simply illustrate that the evolution of the attributes a_t and b_t use standard tools from simulation.

There is a similar function for the train transition function. For example, it is through the train transition function that we represent the fact that a train may consist of three segments. Thus, the train schedule may have trains going from A to B, B to C, and C to D, but if this is really the same train, we have to capture the fact that the train from B to C cannot move until the train from A to B has arrived (as well as modeling any delays that may have incurred).

Given the attribute transition functions $a^M(a, d, W)$ and $b^M(b, d, W)$, we can construct the incidence functions

$$\delta^R_{t+1,a',ad}(\omega) = \begin{cases} 1 & \text{If } a' = a^M(a_t, d_t, W_{t+1}(\omega)), \\ 0 & \text{otherwise,} \end{cases}$$

$$\delta^D_{t+1,b',bd}(\omega) = \begin{cases} 1 & \text{If } b' = b^M(b_t, d_t, W_{t+1}(\omega)), \\ 0 & \text{otherwise.} \end{cases}$$

We can then use these functions to model the transition of the locomotive and train vectors R_t and D_t using

$$R_{t+1,a'} = \sum_{a\in\mathscr{A}} \sum_{d\in\mathscr{D}^R} \delta^R_{t+1,a',ad}(\omega) x^R_{tad},$$

$$D_{t+1,b'} = \sum_{b\in\mathscr{B}} \sum_{d\in\mathscr{D}^D} \delta^D_{t+1,b',bd}(\omega) x^D_{tbd}.$$

We note that although this is notationally useful, in software, the transition function is handled directly by the attribute transition functions applied to each locomotive and train once the decisions have been made.

Appendix C. The Constraints at Time t

The constraints represented by $x_t \in \mathscr{X}_t$ capture flow conservation, along with the more constraints governing the process of consist formation. Basic flow conservation constraints include

$$\sum_{d\in\mathscr{D}^R} x_{tad} = R_{ta}, \quad \sum_{d\in\mathscr{D}^D} x_{tbd} = D_{tb}.$$

In addition, x^R_t and x^D_t must be integer. For the multicommodity and multiattribute models, the flows must be zero or one. For the single commodity problem, they may be greater than one, but still integer.

The more complex constraints involve the operational constraints associated with assigning locomotives to trains. A train cannot move until the minimum power requirement is met. Even though trains can move with minimum power, planners prefer to assign the number of locomotives that fulfills the goal requirement. The units of power added to a train above the goal requirement are assessed the repositioning cost. The total power assigned to a train should not exceed the maximum power allowed on the train.

To model these constraints, we introduce the following:

w_{tb} = the delay imposed on train $b \in \mathscr{B}$ at time t due to the specific assignment of individual locomotives;

y^E_{tb} = the number of power units added on a train $b \in \mathscr{B}$ at time t for repositioning;

y^G_{tb} = the number of power units assigned to a train $b \in \mathscr{B}$ at time t to fulfill the goal requirement;

z_{tb} = the number of locomotive consist splits at time t, for the consist identified by train with attribute b, for example, if a four-locomotive consist is broken into a consist with two locomotives, along with two separate locomotives, this would produce $z_{tb} = 2$.

We use y^G_{tb} to denote the power units dedicated to pull train b and y^E_{tb} to denote the power units added to b for repositioning. We also let a_p denote the power units provided by one locomotive $a \in \mathscr{A}$, which is equal to either a_6 or to a_7, depending on what type of power unit is used by the railroad (axles or horsepower). If $\bar{\mathscr{A}}$ represents the set of dead locomotives that need to be pulled to the shop, then the minimum and maximum power requirement constraints for trains with attributes $b \in \mathscr{B}$ can be written as follows:

$$\sum_{a\in\mathscr{A}\setminus\bar{\mathscr{A}}} \sum_{d\in\mathscr{D}^D \mid d_b = b} a_p x^R_{tad} \geq b_{12} x^D_{tb}, \tag{C1}$$

$$\sum_{a\in\mathscr{A}} \sum_{d\in\mathscr{D}^D \mid d_b = b} a_p x^R_{tad} \leq b_{14} x^D_{tb}. \tag{C2}$$

The goal power and the repositioned power constraints can be expressed using the following inequalities:

$$y^G_{tb} \leq \inf\left(\left[\sum_{a\in\mathscr{A}\setminus\bar{\mathscr{A}}} \sum_{d\in\mathscr{D}^D \mid d_b = b} a_p x^R_{tad}\right] - b_{12} x^D_{tb}, b_{13} - b_{12}\right), \tag{C3}$$

$$y^E_{tb} \geq \left[\sum_{a\in\mathscr{A}} \sum_{d\in\mathscr{D}^D \mid d_b = b} a_p x^R_{tad}\right] - b_{12} - y^G_{tb}, \tag{C4}$$

$$y^G_{tb} \geq 0, \tag{C5}$$

$$y^E_{tb} \geq 0. \tag{C6}$$

As trains move in the network, they come across areas enforcing special regulations requiring that each train must be powered by a number of locomotives with special equipment such as the *locomotive speed limiter*, the *cab signaling system*, and the *flush toilet system*. Notice that the LSL equipment fulfills the CSS and the FTS requirements, and the CSS equipment fulfills the FTS requirements. The following constraint ensures that trains of attributes $b \in \mathscr{B}$ cannot move unless the special equipment requirements are satisfied:

$$\sum_{a\in\mathscr{A}\setminus\bar{\mathscr{A}} \mid a_{14} = b_{15}} \sum_{d\in\mathscr{D}^D \mid d_b = b} x^R_{tad} \geq b_{16} x^D_{tb}, \quad d \in \mathscr{D}^D \mid b_d = b. \tag{C7}$$

Since not all locomotives and trains are available at the beginning of time period t, we need to quantify the train delay w_{tb} of each train $b \in \mathscr{B}$ associated with each feasible solution $x_t \in \mathscr{X}_t$ factored into the objective function contribution for period t defined by (6); w_{tb} is computed by solving the following constraints:

$$(\bar{a}_8 - b_9) x^R_{tad} \leq w_{tb}, \quad \forall\, a \in \mathscr{A}, \forall\, d \in \mathscr{D}^D \mid b_d = b, \tag{C8}$$

$$w_{tb} \geq 0, \tag{C9}$$

where $\bar{a}_1$ is the adjusted available time of the locomotive a. If the locomotive a is still attached to its inbound train $b' \neq b$, then $\bar{a}_8$ is set to a_8 plus a set-off time. Otherwise, $\bar{a}_8 = a_8$.

When a number of locomotives are assigned to a train, setting these locomotives together as an entity named *consist* requires time and manpower. To minimize both train delays and operational costs, it is important to avoid breaking consists

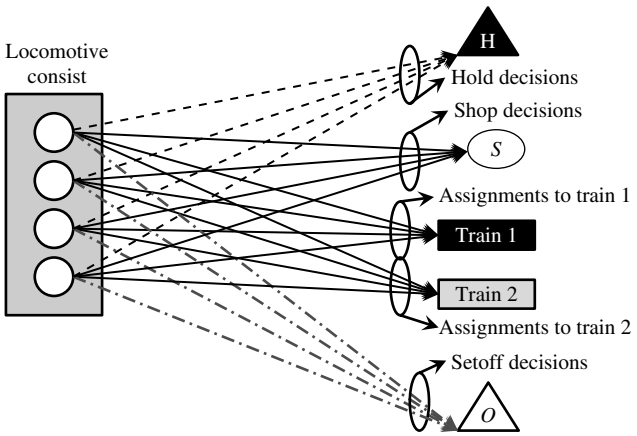

Figure C.1 Locomotive Consist Break-Up Illustration

arriving at each yard to form new ones to power the outbound trains. To model this problem, we use $\mathscr{A}^k$ to denote the set of locomotive attribute vectors in a consist $k \in \mathscr{K}_t$ and R^k_{ta} to denote the number of locomotives of attributes $a \in \mathscr{A}^k$. As shown by Figure C.1, each locomotive of a consist k can be used in one of the following four ways: hold to the next period, enter to a shop for either maintenance or repair, and assign to a train. A consist k (with more than one locomotive) would be segmented in at least two parts if the locomotives in $\mathscr{A}^k$ are utilized in at least two different ways among the four just described. To determine the number of consist splits, which we denote by z_{tk}, we introduce the following binary variables:

z^H_{tk} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k$ is held to the next period and 0 otherwise;
z^S_{tk} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k$ is set-off from k and 0 otherwise;
z^M_{tk} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k$ is entered to shop for maintenance, and 0 otherwise;
z^{Rep}_{tk} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k$ is entered to shop for repair and 0 otherwise;
z^B_{tkb} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k$ is assigned to a train $b \in \mathscr{B}$ and 0 otherwise;
M = an arbitrary large number greater than or equal to $\sum_{a\in\mathscr{A}^k} R^k_{ta}$.

We now write the consist constraints to bound z_{tk} given a feasible solution $x_t \in \mathscr{X}_t$:

$$\sum_{a\in\mathscr{A}^k}\sum_{d\in\mathscr{D}^H} x^R_{tad} \le M z^H_{tk}, \tag{C10}$$

$$\sum_{a\in\mathscr{A}^k}\sum_{d\in\mathscr{D}^S} x^R_{tad} \le M z^S_{tk}, \tag{C11}$$

$$\sum_{a\in\mathscr{A}^k}\sum_{d\in\mathscr{D}^M} x^R_{tad} \le M z^M_{tk}, \tag{C12}$$

$$\sum_{a\in\mathscr{A}^k}\sum_{d\in\mathscr{D}^{Rep}} x^R_{tad} \le M z^{Rep}_{tk}, \tag{C13}$$

$$\sum_{a\in\mathscr{A}^k}\sum_{d\in\mathscr{D}^D \mid d_b=b} x^R_{tad} \le M z^B_{tkb}, \quad \forall\, b \in \mathscr{B}, \tag{C14}$$

$$z^H_{tk} + z^S_{tk} + z^M_{tk} + z^{Rep}_{tk} + \sum_{b\in\mathscr{B}} z^B_{tkb} - 1 \le z_{tk}, \tag{C15}$$

$$z^H_{tk}, z^S_{tk}, z^M_{tk}, z^{Rep}_{tk}, z^B_{tkb} \ge 0, \quad \text{and binary}, \tag{C16}$$

$$z_{tk} \ge 0, \quad \text{and integer}. \tag{C17}$$

Appendix D. Foreign Interchange and Shop Routing

There are more than 300 freight railroads in North America. To improve productivity, these railroads share many of their assets on a daily basis. In this paper, we model the locomotive interchange process to minimize the leasing cost and maintain a balanced fleet over time by controlling the number of locomotives assigned to trains leaving the network, given the current excess of foreign locomotives. For a foreign railroad $f \in \mathscr{F}$ we let $\mathscr{A}^f$ be the set of locomotives owned by f and $\mathscr{B}^f$ be the set of trains leaving to f. The excess of locomotives owned by f at time period t, denoted by R^F_{tf}, is bounded by the following constraints:

$$\sum_{a\in\mathscr{A}^f}\sum_{d\in\mathscr{D}^D \mid d_b\in\mathscr{B}^f} x^R_{tad} - \sum_{a\in\mathscr{A}^f} R^A_{ta} \le \bar{R}^F_{tf}, \tag{D1}$$

$$\bar{R}^F_{(t-1)f} + \sum_{a\in\mathscr{A}^f} R^A_{ta} - \sum_{a\in\mathscr{A}^f}\sum_{d\in\mathscr{D}^D \mid d_b\in\mathscr{B}^f} x^R_{tad} \le R^F_{tf}, \tag{D2}$$

$$\bar{R}^F_{tf}, R^F_{tf} \ge 0. \tag{D3}$$

By calibrating the leasing cost c^F applied to variable R^F_{tf} in the contribution function (6), the model will minimize the excess of foreign locomotives in the network over time.

We now formulate the constraints related to shop routing. Locomotives need to be regularly inspected and repaired. For this, the locomotive planners always try to route locomotives to a shop while they are still useful. This is done by assigning the units that are due soon for maintenance or repair to trains that bring them closer to a shop. The Federal Railroad Administration (FRA) mandates that each locomotive has to be inspected within 92 days from the last maintenance date. When the maintenance due date occurs before arriving to a shop, the engine is switched off and the unit is towed to the nearest shop. Whereas maintenance events are scheduled and known in advance, the repair events occur randomly. A random failure puts the locomotive in a state where either it can only be towed to a shop or it can be used to pull trains for a number of days before it turns dead. A number of factors are taken into account to decide when and where a locomotive will be serviced. Among these are the following:

- the locomotive current state (dead or operational) and the type of shop work required;
- the date by which the locomotive needs to be in a shop;
- the shop congestion level at the maintenance and repair locations.

To formulate the shop routing problem constraints of time period t, we first need to determine the congestion level at each shop $s \in \mathscr{S}$. To do this, we introduce the following additional notation:

$\tau_s(W)$ = the number of time periods required to service one locomotive at shop $s \in \mathscr{S}$;
κ_s = the servicing capacity at shop $s \in \mathscr{S}$.

Assume that we have a feasible solution $x \in \mathscr{X}$, we want to calculate the number of locomotives at each shop at each period t that we denote by R^{Shp}_{ts}. We start by calculating the

number of locomotives sent to a shop $s \in \mathcal{S}$ at each time period t, x_{ts}, as follows:

$$x_{ts}^{Shp} = \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}^{Shp}} x_{tad}^{R}. \tag{D4}$$

Now, assume that the locomotives are serviced at every shop $s \in \mathcal{S}$ following the first-in, first-out policy and the service duration is deterministic ($\tau_s(W) = \tau_s$), then the following must hold at every period t:

$$\xi_{ts} = \begin{cases} \min(q_{(t-1)s} + x_{ts}, \kappa_s - \vartheta_{(t-1)s} + \xi_{(t-\tau_s)s}), \\ \quad \text{if } (\vartheta_{(t-1)s} - \xi_{(t-\tau_s)s} + 1) \le \kappa_s, \\ 0, \quad \text{otherwise}, \end{cases} \tag{D5}$$

$$\vartheta_{ts} = \vartheta_{(t-1)s} - \xi_{(t-\tau_s)s} + \xi_{ts}, \tag{D6}$$

$$q_{ts} = q_{(t-1)s} + x_{ts} - \xi_{ts}, \tag{D7}$$

$$\delta_{ts} = \vartheta_{(t-1)s} - \xi_{(t-\tau_s)s} + q_{(t-1)s}, \tag{D8}$$

$$R_{ts}^{Shp} = \delta_{ts} + x_{ts}. \tag{D9}$$

The variables used in the above equations are
ξ_{ts} = the number of locomotives starting service;
ϑ_{ts} = the number of locomotives being serviced;
q_{ts} = the queue length;
δ_{ts} = the number of locomotives at the shop before adding x_{ts}.

The nonlinear congestion cost $c_{ts}^{ShpCong}(R_{ts}^{Shp})$ added to the problem objective function should minimize the shop congestion over time.

Appendix E. The PLASMA/SC Planning Model

In this model, the time discretization is done at the day level ($T = 7$). We use the notation introduced in §3 to describe this model with the following minor changes and additions:

$\mathcal{A}_t^S$: the set of locomotives with attributes $a \in \mathcal{A}$ released from shop at the beginning of day t;
$\mathcal{A}_t^B$: the set of locomotives with attributes $a \in \mathcal{A}$ becoming available at day t after moving a train;
$\mathcal{A}_t = \mathcal{A}_t^S \cup \mathcal{A}_t^B$;
$\mathcal{A}^S = (\mathcal{A}_t^S)_{t \in \{0,\dots,T\}}$;
$\mathcal{A}^B = (\mathcal{A}_t^B)_{t \in \{0,\dots,T\}}$;
$\mathcal{A} = (\mathcal{A}_t)_{t \in \{0,\dots,T\}}$;
$\mathcal{D}_t^S$: the set of decisions of sending locomotives to shops by the end of day t; here is one decision $d_s \in \mathcal{D}_t^M$ per shop $s \in \mathcal{S}$ ($|\mathcal{D}_t^M| = |\mathcal{S}|$) for each locomotive node;
R_{tas}^S: the number of locomotives with attributes $a \in \mathcal{A}^S$ out of shop $s \in \mathcal{S}$ at the beginning of day t (given as input to the model);
R_{tas}^B: the target number of locomotives with attributes $a \in \mathcal{A}^B$ the model tries to send to shop $s \in \mathcal{S}$ by the end of day t (given as input to the model);
$a^M(a, d)$: the transition function defined by (1) in a deterministic environment;
$b^M(b, d)$: the transition function defined by (2) in a deterministic environment;
x_{tb}^E: an integer (binary) variable representing the number of trains of attributes $b \in \mathcal{B}_t$ moved by the model and used to reposition power;
x_t^E: $(x_{tb}^E)_{b \in \mathcal{B}_t}$;
x_t: (x_t^R, x_t^D, x_t^E).

For each locomotive with attributes $a \in \mathcal{A}$, the attributes a_3, a_4, a_{11}, and a_{12} are dropped (aggregated to a common and meaningless attribute), and we set $a_1 = a_8$. We let $\Gamma^-(t, \tau)$ be the function returning the set of backward ordered $\tau + 1$ time periods ($0 \le \tau \le T - 1$) starting at t. Since in this model time is cyclic (see Figure 2), this function can be explicitly stated as follows:

$$\Gamma^-(t, \tau) = \begin{cases} \{t - \tau, \dots, t\}, & \text{if } t \ge \tau, \\ \{T - \tau, \dots, T, 0\}, & t = 0, \\ \{T + t - \tau, \dots, T, 0, \dots, t\}, & \text{otherwise.} \end{cases}$$

In this problem, we assume that all locomotives sent to shop using a decision $d \in \mathcal{D}_t^S$ leave the system. This can be achieved by setting the duration of d equals to more than seven days ($\tau_d > 7$ days, for all $d \in \mathcal{D}_t^S$). However, if a train or a locomotive is acted on with a decision $d \in \mathcal{D}_t \backslash \mathcal{D}_t^S$ with an ending time greater than the planning horizon, then the time is mapped within the horizon. Assuming that the duration of each decision $d \in \mathcal{D}_t \backslash \mathcal{D}_t^S$ is shorter than seven days, the mapping of the locomotive and train time attributes is done as follows:

$$a_i' = \begin{cases} a_i' - T, & \text{if } a_i' \ge T \\ a_i', & \text{otherwise,} \end{cases}$$

$$\forall (a', a) \in \mathcal{A}^B \times \mathcal{A}^B, \forall d \in \mathcal{D} \backslash \mathcal{D}^S \mid a^M(a, d) = a', \tag{E1}$$

$$b_i' = \begin{cases} b_i' - T, & \text{if } b_i' \ge T \\ b_i', & \text{otherwise,} \end{cases}$$

$$\forall (b', b) \in \mathcal{B} \times \mathcal{B}, \forall d \in \mathcal{D}^D \mid b^M(b, d) = b'. \tag{E2}$$

Now, we formulate the problem constraints. We start with the flow conservation constraints involving the locomotives with attributes $a \in \mathcal{A}_t^S$ coming out of a shop at day t:

$$\sum_{d \in \mathcal{D}_t^S} x_{tad}^R = 0, \quad \forall a \in \mathcal{A}_t^S, \tag{E3}$$

$$\sum_{d \in \mathcal{D}_t^D} x_{tad}^R = R_{tas}^S, \quad \forall a \in \mathcal{A}_t^S. \tag{E4}$$

To route the required number of locomotives to the shops by the end of day t, we add the following constraints:

$$\sum_{t' \in \Gamma^-(t, 6)} \sum_{a \in \mathcal{A}_{t'}^B \mid a_2 = s} x_{t'ad_s}^R \le R_{tas}^B, \quad \forall s \in \mathcal{S}. \tag{E5}$$

For the locomotives with $a \in \mathcal{A}_t^B$, the flow conservation constraints are expressed by the equation:

$$\sum_{t' \in \Gamma^-(t, 6)} \sum_{a' \in \mathcal{A}_{t'}} \sum_{d \in \mathcal{D}_{t'}^D \mid a^M(a', d) = a} x_{t'a'd}^R = \sum_{d \in \mathcal{D}_t} x_{tad}^R, \quad \forall a \in \mathcal{A}_t^B. \tag{E6}$$

A locomotive with attributes $a \in \mathcal{A}_t$ is assignable to a train $b \in \mathcal{B}_{t'}$ ($x_{tad_b}^B$ is feasible) if the locomotive and the train are at the same location ($a_2 = b_2$) and, when the locomotive is not on a pass-through train, the time separating the locomotive availability and the train departure, that we denote by $H(a, b)$, must be contained within a given time interval

$[\underline{h}(a_2), \bar{h}(a_2)]$ at the assignment location. In this model where time cyclic, $H(a,b)$ can be calculated as follows:

$$H(a,b) = \begin{cases} b_1 - (a_1 + O(a_2)), & \text{if } a_1 < b_1 \\ b_1 + T - (a_1 + O(a_2)), & \text{if } a_1 > b_1 \\ -O(a_2), & \text{otherwise,} \end{cases}$$
$$\forall\, (a,b) \in \mathscr{A}^B \times \mathscr{B}. \quad \text{(E7)}$$

The function $O(a_2)$ returns the set-off time of the yard a_2 where the assignment takes place (typically four to six hours) if a setoff is required. As in the detailed model, the trains power requirements must be enforced:

$$\sum_{t' \in \Gamma^-(t,6)} \sum_{a \in \mathscr{A}_{t'}} a_6 x^R_{t'ad} \geq b_{12} x^D_{tb}, \quad \forall\, d \in \mathscr{D}^D \mid d_b = b, \quad \text{(E8)}$$

$$\sum_{t' \in \Gamma^-(t,6)} \sum_{a \in \mathscr{A}_{t'}} a_6 x^R_{t'ad} \leq b_{14} x^D_{tb}, \quad \forall\, d \in \mathscr{D}^D \mid d_b = b. \quad \text{(E9)}$$

In this model, the power repositioning is allowed only on trains starting from shop locations. To achieve this, we limit the power assigned from inbound trains to the outbound train goal requirement:

$$\sum_{t' \in \Gamma^-(t,6)} \sum_{a \in \mathscr{A}^B_{t'}} a_6 x^R_{t'ad} \leq b_{13} x^D_{tb}, \quad \forall\, d \in \mathscr{D}^D \mid d_b = b, \quad \text{(E10)}$$

$$\sum_{t' \in \Gamma^-(t,6)} \sum_{a \in \mathscr{A}^S_{t'}} a_6 x^R_{t'ad} \leq (b_{14} - b_{13}) x^E_{tb}, \quad \forall\, b \in \mathscr{B}_t. \quad \text{(E11)}$$

Multisegment trains have to run according to a sequencing order as earlier train segments have to move first. We formulate this constraint with the following equation:

$$x^D_{t'b'} \leq x^D_{tb}, \quad \forall\, (b,b') \in \mathscr{B} \times \mathscr{B} \mid b' = b_{20}. \quad \text{(E12)}$$

The consist breakup constraints in the single commodity model are similar to the other models. To formulate these constraints, we use notation similar to the one introduced in Appendix C:

$\mathscr{A}^k_t$ = set of possible locomotive attributes in consists $k \in \mathscr{K}$ at day t;
z^S_{tk} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k_t$ is sent to shop, and 0 otherwise;
z^B_{tkb} = a variable that equals 1 when at least one locomotive from $\mathscr{A}^k_t$ is assigned to a train $b \in \mathscr{B}$ and 0 otherwise.

The constraints bounding the number of breakups z_{tk} associated with each consist $k \in \mathscr{K}$ can now be formulated as follows:

$$\sum_{a \in \mathscr{A}^k_t} \sum_{d \in \mathscr{D}^S} x^R_{tad} \leq M z^S_{tk}, \quad \text{(E13)}$$

$$\sum_{a \in \mathscr{A}^k_t} x^R_{tad} \leq M z^B_{tkb}, \quad \forall\, (b,d) \in \mathscr{B} \times \mathscr{D}^D \mid d_b = b, \quad \text{(E14)}$$

$$z^S_{tk} + \sum_{b \in \mathscr{B}} z^B_{tkb} - 1 \leq z_{tk}, \quad \text{(E15)}$$

$$z^S_{tk}, z^B_{tkb} \geq 0, \quad \text{and binary}, \quad \text{(E16)}$$

$$z_{tk} \geq 0, \quad \text{and integer}. \quad \text{(E17)}$$

The most challenging constraints of this problem are the ones related to the solution repeatability over the weekdays at the train symbol level. Some of the train schedules are duplicated over some days of the week and it is important to power those trains the same way (at the train symbol level) during the week. All trains of the same repeatable schedule share the same train symbol. We use $\mathscr{Y}$ to denote the set of possible train symbols. We also let $\mathscr{B}^y$ represent the set of possible train attributes with the same symbol $y \in \mathscr{Y}$ ($\forall\, b \in \mathscr{B}^y, b_{23} = y$). For each train $b \in \mathscr{B}^y$, we calculate the number of locomotives assigned to b from inbound trains with symbol $y' \in \mathscr{Y}$, that we denote by $x^{\mathscr{Y}}_{y'b}$, as follows:

$$\sum_{t' \in \Gamma^-(t,6)} \sum_{a \in \mathscr{A}^B_{t'y'}} \sum_{d \in \mathscr{D}^D \mid d_b = b} x^R_{t'ad} = x^{\mathscr{Y}}_{y'b}, \quad \forall\, b \in \mathscr{B}^y. \quad \text{(E18)}$$

In the above equation, the set $\mathscr{A}^B_{t'y'}$ contains the locomotives with attributes $a \in \mathscr{A}^B_{t'}$ after moving a train with symbol $y' \in \mathscr{Y}$ ($a \in \mathscr{A}^B_{t'} \mid a'_{13} = y'$). The goal here is to keep the vector $x^{\mathscr{Y}}_b = (x^{\mathscr{Y}}_{y'b})_{y' \in \mathscr{Y}}$ as stable as possible across all the trains in $\mathscr{B}^y$. For each pair of trains with attributes $(b,b') \in \mathscr{B}^y \times \mathscr{B}^y$, the change in number of locomotives obtained from trains with symbol $y' \in \mathscr{Y}$ is obtained by

$$\epsilon_{bb'y'} = |x^{\mathscr{Y}}_{y'b} - x^{\mathscr{Y}}_{y'b'}|, \quad \forall\, (b,b') \in \mathscr{B}^y \times \mathscr{B}^y, \forall\, y' \in \mathscr{Y}.$$

To minimize the changes across all pairs of trains, we add a cost to each variable $\epsilon_{bb'y'}$, and rewrite the above constraint as follows:

$$x^{\mathscr{Y}}_{y'b} - x^{\mathscr{Y}}_{y'b'} \leq \epsilon_{bb'y'}, \quad \text{(E19)}$$

$$x^{\mathscr{Y}}_{y'b'} - x^{\mathscr{Y}}_{y'b} \leq \epsilon_{bb'y'}, \quad \text{(E20)}$$

$$\epsilon_{bb'y'} \geq 0. \quad \text{(E21)}$$

We use the following costs and rewards in the objective function:

c^D_{tb}: the contribution earned from moving one train $b \in \mathscr{B}_t$ (c^D_{tb} is positive b if it is a regular train and negative if b is a light engine);
c^E_{tb}: the cost of using train $b \in \mathscr{B}_t$ to reposition power (an "empty" movement);
c^R_{tad}: the contribution earned from acting on one locomotive $a \in \mathscr{A}_t$ with a decision $d \in \mathscr{D}$;
c^K: the consist breakup cost;
c^Y: the cost of having one change in the solution aggregated to the train symbol level over time.

To enforce a FIFO policy when assigning locomotives to trains, we use the following locomotive-train assignment contribution for each decision $d \in \mathscr{D}^D$:

$$c^R_{tad} = \begin{cases} 0, & \text{if } \underline{h}(a_2) \leq H(a,b) \leq h^*(a_2) \\ -\beta |H(a,d_b) - h^*(a_2)|^\alpha, & \text{if } h^*(a_2) \leq H(a,b) \leq \bar{h}(a_2) \\ -\infty, & \text{otherwise,} \end{cases}$$

where $H(a,b)$ is as defined by (E7), $[\underline{h}(a_2), \bar{h}(a_2)]$, and $[\underline{h}(a_2), h^*(a_2)]$ are the feasible and the best connection time interval at the assignment location a_2, and α and β are positive parameters ($\alpha > 1$). The single commodity planning model can now be formulated as follows:

$$\arg\max_{x \in \mathscr{X}} \Bigg\{ \sum_{t=0}^{T} \sum_{b \in \mathscr{B}_t} (c^D_{tb} x^D_{tb} + c^E_{tb} x^E_{tb}) + \sum_{t=0}^{T} \sum_{a \in \mathscr{A}_t} \sum_{d \in \mathscr{D}} c^R_{tad} x^R_{tad} + c^K \sum_{t=0}^{T} \sum_{k \in \mathscr{K}_t} z_{tk} + c^Y \sum_{y \in \mathscr{Y}} \sum_{y' \in \mathscr{Y} \setminus \{y\}} \sum_{b \in \mathscr{B}^y} \sum_{b' \in \mathscr{B}^y \setminus \{b\}} \epsilon_{bb'y'} \Bigg\}, \quad \text{(E22)}$$

where $\mathscr{X}$ is the set of feasible solutions $x = (x_t)_{t \in \{0,\ldots,T\}}$ as defined by constraints (E3)–(E21).

References

Ahuja R, Cunha C, Sahin G (2005) Network models in railroad planning and scheduling. *TutORials in Operations Research* (INFORMS, Catonsville, MD).

Ahuja R, Magnanti TL, Orlin J (1993) *Network Flows: Theory, Algorithms, and Applications*, Vol. 91 (Prentice Hall, Englewood Cliffs, NJ).

Ahuja RK, Liu J, Orlin JB, Sharma D, Shughart LA (2005) Solving real-life locomotive-scheduling problems. *Transportation Sci.* 39(4):503–517.

Cordeau J-F, Soumis F, Desrosiers J (2000) A Benders decomposition approach for the locomotive and car assignment problem. *Transportation Sci.* 34(2):133–149.

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

Cordeau J-F, Toth P, Vigo D (1998) A survey of optimization models for train routing and scheduling. *Transportation Sci.* 32(4): 380–404.

Crainic T, Rousseau J-M (1988) Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem. *Transportation Res. Part B* 20(3):290–297.

Crainic TG, Laporte G (1997) Planning models for freight transportation. *Eur. J. Oper. Res.* 97(3):409–439.

Dejax P, Crainic T (1987) A review of empty flows and fleet management models in freight transportation. *Transportation Sci.* 21(4):227–247.

Glickman T, Sherali H (1985) Large-scale network distribution of pooled empty freight cars over time, with limited substitution and equitable benefits. *Transportation Res. Part B* 19(2):85–94.

Glover F, Kochenberger G (2003) *Handbook of Metaheuristics* (Springer, Norwell, MA).

Godfrey G, Powell WB (2001) An adaptive, distribution-free algorithm for the newsvendor problem with censored demands, with applications to inventory and distribution. *Management Sci.* 47(8):1101–1112.

Godfrey G, Powell WB (2002) An adaptive, dynamic programming algorithm for stochastic resource allocation problems I: Single period travel times. *Transportation Sci.* 36(1):21–39.

Haghani A (1989) Formulation and solution of a combined train routing and makeup, and empty car distribution model. *Transportation Res. Part B* 23(6):433–452.

Herren H (1977) Computer controlled empty wagon distribution on the SSB. *Rail Internat.* 8(1):25–32.

Holmberg K, Joborn M, Lundgren JT (1998) Improved empty freight car distribution. *Transportation Sci.* 32(2):163–173.

Joborn M (2001) Optimization of empty freight car distribution in scheduled railways. Ph.D. thesis, Department of Mathematics, Linkoping University, Linkoping, Sweden.

Joborn M, Crainic TG, Gendreau M, Holmberg K, Lundgren JT (2004) Economies of scale in empty freight car distribution in scheduled railways. *Transportation Sci.* 38(2):121–134.

Jordan W, Turnquist M (1983) A stochastic dynamic network model for railroad car distribution. *Transportation Sci.* 17(2):123–145.

Lingaya N, Cordeau J-F, Desaulniers G, Desrosiers J, Soumis F (2002) Operational car assignment at via Rail Canada. *Transportation Res. Part B* 36(9):755–778.

Mendiratta VB, Turnquist MA (1982) A model for the management of empty freight cars. *Transportation Res. Record* 838:50–55.

Nemani A, Ahuja R (2011) OR models in freight railroad industry. *Wiley Encyclopedia of Operations Research and Management Science* (John Wiley & Sons, Hoboken, NJ).

Powell WB (2007) *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (John Wiley & Sons, New York).

Powell WB (2011) *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. (John Wiley & Sons, Hoboken, NJ).

Powell WB, Bouzaiene-Ayari B, Simão H (2006) Dynamic models for freight transportation. Laporte G, Barnhart C, eds. *Handbooks in Operation Research and Management Science: Transportation*, Vol. 14 (North-Holland, Amsterdam), 285–366.

Powell WB, Jaillet P, Odoni A (1995) Stochastic and dynamic networks and routing. Monma C, Magnanti T, Ball M, eds. *Handbook in Operations Research and Management Science, Volume on Networks* (North-Holland, Amsterdam), 141–295.

Powell WB, Shapiro JA, Simão HP (2001) A representational paradigm for dynamic resource transformation problems. Coullard RFC, Owens JH, eds. *Annals of Operations Research* (J.C. Baltzer AG, Basel, Switzerland), 231–279.

Powell WB, Shapiro JA, Simão HP (2002) An adaptive dynamic programming algorithm for the heterogeneous resource allocation problem. *Transportation Sci.* 36(2):231–249.

Simão HP, Day J, George A, Gifford T, Powell WB, Nienow J (2009) An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Sci.* 43(2):178–197.

Topaloglu H, Powell WB (2006) Dynamic programming approximations for stochastic, time-staged integer multicommodity flow problems. *INFORMS J. Comput.* 18(1):31–42.

Vaidyanathan B, Ahuja R (2008) Real-life locomotive planning: New formulations and computational results. *Transportation Res. Part B* 42(2):147–168.

Vaidyanathan B, Ahuja R, Orlin JB (2008) The locomotive routing problem. *Transportation Sci.* 42(4):492–507.

White W (1972) Dynamic transshipment networks: An algorithm and its application to the distribution of empty containers. *Networks* 2(3):211–236.

Ziarati K, Soumis F, Desrosiers J, Solomon M (1999) A branch-first, cut-second approach for locomotive assignment. *Management Sci.* 45(8):1156–1168.

Ziarati K, Soumis F, Desrosiers J, Gelinas S, Saintonge A (1997) Locomotive assignment with heterogeneous consists at CN North America. *Eur. J. Oper. Res.* 97(2):281–292.