Approximate dynamic programming for locomotive optimization

Warren B. Powell
Belgacem Bouzaiene-Ayari
Department of Operations Research and Financial Engineering
Princeton University, Princeton, NJ 08544

October 28, 2006

For over 10 years we have been developing a series of models for optimizing locomotives for a major freight railroad in the U.S. using the principles of approximate dynamic programming. The projects span operational planning to strategic planning which generally impose very different expectations in terms of the level of realism. In this talk, we review how these projects unfolded and the surprising level of detail that was required to produce implementable results, even for a strategic system.

The foundation of our solution strategy is approximate dynamic programming, which combines the flexibility of simulation with the intelligence of optimization. In our prior work, we have focused on the power of ADP to handle uncertainty, overcoming classical computational problems associated with techniques such as Markov decision processes or stochastic programming. What is easy to overlook is that ADP is basically intelligent simulation, and as such it is able to handle an extremely high level of detail (effectively, anything a simulation model can handle). Surprisingly, ADP has also enabled us to use Cplex to solve integer programs that arise in locomotive optimization that have been thought to be solvable only with heuristics.

In this talk, we discuss the limitations of classical optimization models of fleet management, focusing not as much on the issue of uncertainty but rather on the importance of capturing realistic operational details. We describe how the ADP paradigm makes it much easier to capture these details, without losing the important features of optimization.

1 Literature review

There is an extensive literature on optimization models for rail operations. These range from single commodity models for managing generic fleets of containers (White & Bomberault (1969), Gorenstein et al. (1971), White (1972), and Herren (1977), to multicommodity models for handling multiple equipment types with substitution (Dejax & Crainic (1987), Cordeau et al. (1998), Crainic & Rousseau (1988), Crainic & Laporte (1997), Powell et al. (1995), Haghani (1989), Joborn (2001), Lingaya et al. (2002) and Joborn et al. (2004)).

Many of these models are particularly well suited for managing fleets of containers (box cars, trailers, intermodal containers). 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 et al. (1998) for a review of the literature as of 1998). 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. Cordeau et al. (2000) and Cordeau et al. (2001) apply Benders decomposition to handle the simultaneous optimization of locomotives and cars. Ahuja et al. (2005) presents a neighborhood search heuristic for a strategic planning model for locomotives which takes into consideration the cost of breaking up sets of locomotives ("consist busting").

2 Modeling issues in locomotive operations

A very common piece of notation in the transportation modeling literature is

x^k_{tij} = The flow of resources of type k leaving node i at time t going to node j.

This notation is at the foundation of virtually any multicommodity flow model. Any model of locomotive operations would use this notation to capture the presence of different types of locomotives. Locomotives differ both in horsepower, which affects their speed, and their "adhesion" (or tractive effort)which affects their ability to get a train moving from a standstill. High-adhesion, high-powered locomotives can move the most freight. The importance of horsepower versus adhesion depends on the grade for a particular train.

When assigning locomotives to trains, the first issue that has to be considered is how much power is needed to pull the train. A train might require 2.2 horsepower per ton. A train weighing 9,000 tons (gross weight, including the weight of the cars), requiring 2.2

horsepower per ton would require enough locomotives to provide 19,800 horsepower. This horsepower can be provided by a mixture of locomotives with anywhere between 3,000 to over 4,000 horsepower. Of course, we have to use an integer number of locomotives, and we can mix and match to produce the right amount of power. Seven 3,000 horsepower locomotives produce 21,000 horsepower, or we could use four 3,000 horsepower units with two 4,000 horsepower units for a total of 20,000 horsepower. As a result, this is a fairly challenging integer programming problem.

If we simply had to schedule a fleet of locomotives taking into consideration the mix of horsepower and integrality requirements, this by itself would be a fairly hard integer programming problem. We also have to consider the fact that if we group multiple locomotives to pull a single train (this group of locomotives is called a *consist*), there is a cost if we have to separate one or more locomotives from the consist. This introduces a significant complication, over and above the challenge of finding an integer number of heterogeneous locomotives to move a train. This complexity motivated the design of the neighborhood search heuristic reported in Ahuja et al. (2005).

Our work has identified a number of other issues which have proven to be important not just for operational models (these tend to be more complex since the results have to capture enough realism for implementation), but also for strategic planning models. These details include:

- Leader locomotives. Each train must have one, and sometimes two, locomotives that are qualified to be the first locomotive in a consist. A train might require two locomotives if the train has to go out and back, and if it is not possible to turn the locomotives around.
- Shop routing. At any point in time, roughly five to seven percent of a locomotive fleet needs to be routed to a shop location for required maintenance.
- Late trains. Most North American railroads have to deal with a level of variation in train arrival times that can be significant.
- New trains and cancellations. New trains may be added at the last minute (coal

trains and grain trains are particularly notorious for last minute requests) or canceled (perhaps due to insufficient traffic).

- Random equipment failures. On any day, a fraction of locomotives may fail, requiring that these be pulled to a shop location, adding to the weight of the train.
- Foreign power. In the U.S., there are two major western railroads and two major eastern railroads. It is common for locomotives from one railroad to pull a train into the yard belonging to another railroad. It is not always the case that the locomotives immediately return.
- Locomotive patterns. Railroads like to have repeatable patterns whenever possible. These patterns tend to be of the form "locomotives arriving to a hard on train A should be put on outbound train B." These patterns arise

Shop routing is particularly difficult. A locomotive can still pull a train while it is being routed to shop, but while we are routing a locomotive toward its shop location, we have to try to minimize how often consists are broken. Shop routing can not be solved independently of the original problem.

In strategic planning applications, it is also important to take into account the random additions and cancellations, as well as delays. If an extra train moves out of a yard 20 percent of the time (to various destinations), then we cannot pretend that we know exactly when, and to where, these additional trains will move.

3 A modeling and algorithmic strategy using ADP

We can handle all of the complexities of these operations using the framework of approximate dynamic programming. We present a model that captures all of these complexities by modeling locomotives as heterogeneous resources (that is, each locomotive is characterized by a vector of attributes), rather than using multicommodity flow notation. We handle the complex physics using a transition function (commonly used in simulation) rather than systems of linear equations. If S_t is the state of our system (the status of all locomotives and trains), x_t is our vector of decisions (for example, the assignment of individual locomotives to trains, decisions to hold locomotives or trains, and the decisions to move foreign power to another railroad), and W_t is a vector of exogenous information (train delays, locomotive failures, new trains and train cancellations), then we model the transition function using

$$S_{t+1} = S^M(S_t, x_t, W_{t+1}).$$

Transition functions are relatively easy to code and can capture virtually unlimited complexity. There is no limit in the complexity of the state variable (for example, modeling a high level of detail in locomotives and trains).

The hard part is learning how to make decisions that optimize at a point in time, as well as over time. A myopic policy would consist of solving

$$x_t = \arg\max_{x_t \in \mathcal{X}_t} C(S_t, x_t) \tag{1}$$

where $\mathcal{X}_t$ captures all the constraints at time t, and $C(S_t, x_t)$ captures all the contributions from decisions made at time t (such as the benefit of moving a train, the cost of assigning certain types of locomotives to certain types of trains, consist-breaking costs, and costs associated with getting power to their shop on time). Equation (1) represents a myopic decision function. It can optimize at a point in time (although the problem is still a fairly difficult integer program), but not over time. We can approximately optimize over time using the following strategy. Recognizing that S_t is the information we need to make a decision (the "pre-decision state variable"), we next define the post-decision state variable S_t^x which is the state at time t immediately after we make a decision. We then break the transition function into two steps:

$$S_t^x = S^{M,x}(S_t, x_t) \tag{2}$$

$$S_{t+1} = S^M(S_t^x, W_{t+1}). \tag{3}$$

We then define $\bar{V}_t(S_t^x)$ as the (approximate) value of being in state S_t^x. We then make decisions using

$$x_t = \arg\max_{x_t \in \mathcal{X}_t} \big(C(S_t, x_t) + \bar{V}_t(S_t^x) \big). \tag{4}$$

where S_t^x is a deterministic function of x_t. If $\bar{V}_t(S_t^x)$ is appropriately defined, equation (4) is no more difficult to solve than (1). We have found that (4) can be solved optimally using Cplex, despite being a fairly large-scale integer program.

ADP solves the problem by stepping through time, using value function approximations to capture the effect of decisions now on the future. Equations (2) and (3) handle the evolution of the system over time. As a result, we approximately solve the problem over time. What was surprising even to us is that by stepping through time, we no longer had to use heuristics to solve the integer programming problem at a point in time. We could handle all of the issues listed above, including consist-breaking, leader locomotives, and routing power to shop, by solving the resulting integer program using Cplex. The research challenge is finding the best value function approximation $\bar{V}$ to optimize decisions over time.

4 Conclusions

We believe this research calls into question the use of classical multicommodity flow models for managing locomotives, since they lack the realism needed to capture actual operations, even for strategic planning purposes such as fleet sizing. Our finding is that approximate dynamic programming combines optimization and simulation, making it possible to handle a virtually unlimited level of detail, while also enabling the ability to provide optimal solutions at a point in time. We gain the ability both to handle a high degree of complexity, as well as the ability to solve a problem at a point in time t to optimality using a commercial solver. The only tradeoff is that we can only approximately optimize over time.

References

Ahuja, R. K., Liu, J., Orlin, J. B., Sharma, D. & Shughart, L. A. (2005), 'Solving real-life locomotive-scheduling problems', *Transportation Science* **39**(4), 503–517. 2, 3

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

Cordeau, J. F., Soumis, F. & Desrosiers, J. (2001), 'Simultaneous assignment of locomotives and cars to passenger trains', *Operations Research* **49**, 531–548. 2

Cordeau, J. F., Toth, P. & Vigo, D. (1998), 'A survey of optimization models for train routing and scheduling', *Transportation Science* **32**, 380–404. 1, 2

Crainic, T. & Rousseau, J.-M. (1988), 'Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem', *Transportation Research B* **20B**, 290–297. 1

Crainic, T. G. & Laporte, G. (1997), 'Planning models for freight transportation', *European Journal of Operational Research* **97**(3), 409–439. 1

Dejax, P. & Crainic, T. (1987), 'A review of empty flows and fleet management models in freight transportation', *Transportation Science* **21**, 227–247. 1

Gorenstein, S., Poley, S. & White, W. (1971), On the scheduling of the railroad freight operations, Technical report 320-2999, ibm philadelphia scientific center, IBM. 1

Haghani, A. (1989), 'Formulation and solution of a combined train routing and makeup, and empty car distribution model', *Transportation Research* **23B**(6), 433–452. 1

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

Joborn, M. (2001), Optimization of empty freight car distribution in scheduled railways, Ph.D. thesis, Department of Mathematics, Linkoping University, Sweden. 1

Joborn, M., Crainic, T. G., Gendreau, M., Holmberg, K. & Lundgren, J. T. (2004), 'Economies of scale in empty freight car distribution in scheduled railways', *Transportation Science* **38**(2), 121–134. 1

Lingaya, N., Cordeau, J. F., Desaulniers, G., Desrosiers, J. & Soumis, F. (2002), 'Operational car assignment at via rail canada', *Transportation Reesarch B* **36**, 755–778. 1

Powell, W. B., Jaillet, P. & Odoni, A. (1995), Stochastic and dynamic networks and routing, *in* C. Monma, T. Magnanti & M. Ball, eds, '*Handbook in Operations Research and Management Science*, Volume on *Networks*', North Holland, Amsterdam, pp. 141–295. 1

White, W. (1972), 'Dynamic transshipment networks: An algorithm and its application to the distribution of empty containers', *Networks* **2**(3), 211–236. 1

White, W. & Bomberault, A. (1969), 'A network algorithm for empty freight car allocation', *IBM Systems Journal* **8**(2), 147–171. 1

Ziarati, K., Soumis, F., Desrosiers, J. & Solomon, M. (1999), 'A branch-first, cut-second approach for locomotive assignment', *Management Science* **45**, 1156–1168. 2