Optimization-based Decision Support System for Train Dispatching

Sifeng Lin
Graduate program of Operations Research and Industrial Engineering,
The University of Texas at Austin, Austin, Texas 78712; sifenglin@utexas.edu

Anant Balakrishnan
McCombs School of Business,
The University of Texas at Austin, Austin, Texas 78712; anantb@mail.utexas.edu

Alper Uygur
BNSF Railway,
2400 Western Center Blvd, Fort Worth, Texas 76131; alper.uygur@bnsf.com

1 Introduction and Problem Description

Increasing the utilization of expensive and scarce rail transport resources—tracks, locomotives, and cars—is of great importance to the railway industry to cope with the increasing customer demand. Effectively dispatching trains on each territory to reduce waiting time and increase velocity is an important tactic to improve the resource utilization, as well as service performance. Every mile-per-hour increase can yield annual savings of millions of dollars in capital and expenses (GE Reports, 2010).

Given the set of trains that will traverse a territory over the planning horizon, and the attributes of each train including its entry location and time, exit location, priority, speed, and track usage restrictions, the train dispatching problem seeks to sequence these trains and plan the meet and pass events so as to maximize average train velocity while satisfying various operational requirements for the trains and tracks. The operational requirements can be listed as follows. (1) Trains in the same direction can trail each other in the same segment in order to improve the utilization of track resources. (2) For safety purposes, trains are required to keep the minimal time or distance separations, called headway, between each other. The trailing headway requires two trailing trains to keep a minimal distance separation, and the control point headway dictates a minimal time separation between two passes through a control point. (4) Some rail tracks may be unavailable to some or all of the trains during a specific time window, due to practical issues including maintenance of way, prescribed train schedules and track usage restrictions. (4) Some types of trains have such a high priority that they should not wait for any other types of trains. (5) Trains may be required to arrive at a specific station within a specific time window. As discussed in Caprara, Fischetti, & Toth(2002), these time windows can be explicit or implicit time windows.

Prior researches have proposed mathematical programming models for train dispatching and attempted to find effective algorithms to solve this problem. However, most models did not fully incorporate all the operational requirements, and few researches study strategies to solve real-life problem instances to optimality or near optimality within reasonable amount of time. Using a discrete-time representation, we propose an integer programming model for the train dispatching that aims at maximizing the total weighted velocity of trains, while taking into account operational requirements including trailing of trains, minimal headway between trains, track unavailability and train priorities. and explore techniques to solve real-life problem instances to optimality.

2 Mathematical Formulation

Similar to Sahin, Ahuja, & Cunha(2008), Caprara, Fischetti, & Toth(2002) and Harrod (2011), we based our model on the discrete time representation. The objective function coefficients are set in such a way that the total weighted velocity is maximized. Our train origin constraint requires trains that are already in the territory to continue their path, but allows trains coming from terminals to wait in terminal if the territory is too congested to accept them. The flow conservation constraints ensures the continuity of train route and the station capacity constraint makes sure that the number of trains moving and waiting in a specific station at any time should not exceed the number of tracks available in the station.

We model the trailing headway and overtaking prevention by enforcing the non-concurrency constraint. The non-concurrency constraint requires that at most one train can appear in each track section in the segment at any time and if any train appears in any section in the segment, then no trains in the opposite direction can appear in the same segment at that time. We model the control point headway by imposing that at most one train can passes through a specific control point during the time window required by the control point headway.

Compared to the overtaking clique constraints discussed in Caprara, Fischetti, & Toth(2002), our way of imposing non-overtaking has two advantages. First, number of constraints are much smaller. Second, by incorportaing more variables in each constraint, a larger clique is constructed, leading to a tighter model.

Our non-concurrency constraint is superior to the track capacity constraints of Sahin, Ahuja, & Cunha (2008) and Harrod (2011) in several aspects. Fisrt, by incorporating the unidirectional movement requirement, which dictates that trains in the opposite direction should never appear in the same segment at the same time, our non-concurrency constraint is tighter than their track capacity constraints. Second, our notion of sections, based on which the non-concurrency constraints is imposed, allows more flexibility to strengthen the model. Third, variables are defined for each segment, instead of each section or block, which reduces number of variables. Finally, the train conflicts because of block occupancy transitions discussed in Harrod (2011) will not occur inside segment by enforcing our non-concurrency constraints.

3 Model Strengthening

The strength of the model is affected by the size of the clique in the non-concurrency constraints, which is impacted by the length of the sections. We then try to refine the sections by elongating them. We provide two guidelines for elongating sections. First, partitions of sections in the same segment in different directions are independent. Second, sections in same segment in a specific direction can overlap with each other. We develop a procedure to define distinct overlapping sections for trains traveling in each direction by exploiting the control point headway. We show that the length of the section should be no shorter than the distance traveled by the second slowest train during the control point headway.

The unidirectional movement property requires that trains in different directions should never appear in the same segment at the same time. The main idea of enforcing the unidirectional movement requirement is to identify cliques to activate the indicator variables. The non-concurrency constraint can be viewed as the section-based unidirectional constraint. We then develop two other sets of the unidirectional inequalities that identify cliques based on the train and the headway respectively. The train-based unidirectional movement inequality makes use of the fact that every train can enter a segment at most once. The headway-based unidirectional movement constraint exploits the property that at most one train can enter a segment during the time window specified by the control point headway. These three sets of unidirectional inequalities complement each other.

By exploiting the FIFO requirement of the dispatching operation, which requires higher priority trains that enter the territory earlier should exit the territory earlier than the lower priority trains that enters the territory later, we show further tighten the train-based unidirectional movement inequality.

4 Sequentially Routing Heuristic

To generate an initial feasible solution to the integer programming solver, we develop a heuristic that sequentially routes the trains and resolves the conflict in time order. To prevent deadlocks, the heuristic makes sure that no station is blocked at the beginning of the each iteration, and trains should not move forward if deadlock may arise; if any deadlock arises, then a lower priority train is reversed to a previous station as a last resort. After making sure that no stations are blocked, the heuristic pick the earliest event and check the feasibility of the event. If the event is not feasible, the corresponding train is required to wait in current station. If the event is feasible, the heuristic resolves the conflicts, if any, associated with that train by applying a greedy rule.

5 Extensions

Based on our mathematical formulation, we can use a hierarchical approach to model the no-wait trains and deal with possible infeasibilities resulting from enforcing explicit time window constraints. (1) To ensure that no-wait trains never wait for other types of trains, we adopt a two stage process that first decides the schedule of no-wait trains, then impose their arrival time at destination as hard constraints and solves for the rest of the trains. (2) To get rid of possible infeasibilities resulting from enforcing explicit time window constraints, we design a three-stage procedure to relax some of the explicit time windows. In the first stage, we solve for the no-wait trains; in the second stage, we impose no-wait trains to arrive at their destination no later than schedule in first stage solution and try to maximize the weighted number of trains that can adhere to their explicit time windows; in the third stage, we impose explicit time windows for trains that can adhere to the time window in second stage, and implicit time windows for all other trains.

6 Computational Results

We test our algorithm on a modified track network and used modified train data from a Class I railroad company in US. The territory is a 100-mile single track territory with 9 stations and the planning horizon is 10 hours. We use 90 seconds as one time unit.

The computational results show that the linear programming upper bound of our model is tighter than that of Harrod (2011), and our section refinement technique and the unidirectional movement inequalities can further improve the linear programming upper bound. The results also demonstrate that adding the train-based unidirectional movement inequality can improve the computational performance substantially. The results also indicate that we can solve many of the problem instances to optimality within reasonable amount of time. Besides, providing the heuristic solution as an initial solution to CPLEX may facilitate the solution process for some of the problem instances, but not all.

7 Conclusion

We propose a novel integer programming model for the train dispatching problem that takes into account various operational considerations including trailing of trains, minimal headway requirements between trains, track unavailability and train priorities. We show that our model is tighter and smaller than discrete time train dispatching models in the literature. To tighten our model, we develop a section refinement procedure by exploiting the control point headway. Besides our non-concurrency constraints, which can be viewed as the section-based unidirectional movement inequality, we propose train-based and headway-based unidirectional inequalities to further improve the computational performance. In order to generate a warm start solution to integer programming solver, we develop a sequential routing heuristic that dispatches trains and resolves conflicts in time order.

Our study may be the first attempt to explore strategies to solve real-life train dispatching problems to optimality or near-optimality by using discrete time representation. Both Caprara, Fischetti, & Toth (2002) and Sahin, Ahuja, & Cunha (2008) end up solving their problems by heuristics, and Harrod (2011) can only apply their model to a small territory with simplified territory data. Our computational results show that our solution method can obtain optimal or near-optimal solution to many real-world problem instances within reasonable amount of time. But as indicated by the computational results, some problems are still too difficult to solve to optimality. Moreover, some of our valid inequalities, although effective theoretically and resulting in a bit tighter bound, do not help the solution process. Developing other strategies to further improve the upper and lower bounds offers promising directions for future researches.

8 Bibliography

Caprara, A., Fischetti, M., & Toth, P. (2002). Modeling and solving the train timetabling problem. *Operations Research*, 851-861.

Dorfman, M. J., & Medanic, J. (2004). Scheduling trains on a railway network using a discrete event model of railway traffic. *Transportation Research Part B*, 81-98.

GE Reports. (2010, June 7). RailEdge tech:Faster, smarter trains to save millions.

Harrod, S. (2011). Modeling network transition constraints with hypergraphs. *Transportation Science*, 81-97.

Sahin, G., Ahuja, R. K., & Cunha, C. B. (2008). Integer programming based approaches for the train dispatching problem.