OPTIMAL RAIL SCHEDULING IN LIMITED FLEXIBILITY ENVIRONMENTS

Danielle Zyngier[*], Jan Lategan and Ludwig Furstenberg
Hatch
Ontario, Canada, L2E7J7

Abstract

Rail scheduling is a challenging problem given both its spatial and temporal characteristics. Rail lines can be hundreds of km long, while single line train crossing strategies are based on a station or passing loop level and require the analysis of the problem on a minute time scale. In mixed-use rail systems with limited passing loop infrastructure, trains have different passing priorities and lengths, thus differing in their ability to use passing loops. Most commercial software tools for simulating rail systems often resort to problem-specific rules and heuristics. They can typically only be used by highly specialized personnel but are still unable to solve complex rail configurations since the simulation approach is not well suited to optimize the train crossing problem. This paper presents an integer formulation for the detailed scheduling of trains on a single main line using the modeling elements/ equations presented as part of the Process Systems-based Unit-Operation-Port-State Superstructure (UOPSS) framework. This model is the basis of the patent-pending Hatch Rail Optimizer (HRO) software. Other approaches in the literature fail to address many of the intricacies solved by our work. This approach is demonstrated through a practical case study involving a 370 km rail corridor with five different train sizes over a week-long scheduling horizon. Interesting computational experiences comparing Mixed Integer Linear Programming (MILP) and Integer Programming (IP) formulations are also discussed.

Keywords

Rail, scheduling, MILP, IP, optimization, UOPSS.

Introduction

In flexible rail networks, trains can cross each other simultaneously through double line sections, and significant yard capacity exists at intermediate stations (where trains may wait). This flexible structure is well suited to the use of discrete event simulation- and/or rule-based scheduling tools, since conflicts between trains can be managed efficiently on a station-to-station basis.

Other rail systems, however, have very limited scheduling flexibility. This occurs when there are long single-line sections interspersed with double line sections. Limitations are worsened when the rail system is used by a combination of freight- and passenger trains ("mixed-use" rail system) with very different train lengths, travelling times and priorities, and an assortment of passing loops lengths, only a few of which can accommodate longer trains.

These limited-flexibility systems are often the result of multiple expansions or upgrades that were implemented over a number of decades. Such systems create a challenge for manual and simulation-/ heuristic-based scheduling, which cannot adequately address the system's complexity while still resulting in a practically useful approach. This results in much lower utilization of rail system capacity that is theoretically available, very slow recovery from

[*] To whom all correspondence should be addressed

unforeseen disruptions in the system such as locomotive breakdowns, as well as difficulties in planning maintenance operations of specific rail sections and deciding on the best infrastructure investment strategies to increase rail system capacity.

Given the capital intensity to expand on existing rail infrastructure, there has been significant interest in rail scheduling problems as a means to identify key track expansion locations, thus increasing business value. This interest in rail scheduling is demonstrated by numerous recent conferences and publications on this topic. Rail scheduling topic was also the recipient of the Edelman prize in 2008 (Kroon *et al*., 2008).

Pellegrini *et al.* (2013) used a continuous-time MILP formulation to achieve the best possible feasible solution within a limited computation time, and obtained interesting insights into solver tuning. Regarding the underlying model, however, there is no mention of passing loops with different lengths, different train sizes nor single- and double-line sections.

More recently, Andersson *et al.* (2015) proposed a MILP approach to increase the robustness of a rail schedule by adjusting the margin times between train launches. Once again, their model does not (1) allow trains to run in both directions, (2) consider multiple train sizes, nor (3) accept the definition of single- and/or double rail line sections.

The next section describes the mathematical formulation of the many elements of the rail system, followed by a case study on a real 370-km mixed-use rail corridor with limited flexibility.

Model Formulation

An illustration of a small section of a single track rail system is shown in Fig. 1. The track between train stations is called a "Section". Passing loops are track segments that allow trains to pass each other.

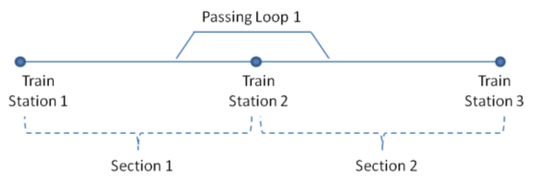

Figure 1. Elements of a single track Rail System

There are several strategies for representing decision-making systems in the literature, such as State-Task Network (STN, Kondili *et al*., 1993), Resource-Task Network (RTN, Pantelides, 1994) and Unit-Operation-Port-State Superstructure (UOPSS, Kelly, 2005; Zyngier and Kelly, 2009). UOPSS was selected to represent the rail system due to its flexibility, ease of representation, and ability to manage selected or limited connectivity between various rail tracks.

In UOPSS, decision-making elements are classified according to their Fill-Hold-Draw (FHD) logistic characteristics. Two very important FHD characteristics are those of batch process units (which introduce a delay in the system) and inventory units (which can accumulate the states that flow through the system). The different elements of the rail system are described in the following sub-sections according to UOPSS terms.

Rail Sections and Passing Loop Representation in UOPSS

Each train enters a rail section, stays within this section for a user-specified travel time (runtime between stations), and then leaves for the next section, freeing up the current section to be used by another train. Due to their FHD characteristics, the sections of a rail line can therefore be represented in UOPSS terms by a series of fixed-time batch (delay) units through which trains ("states") pass:

$$\sum_{tt=0}^{UT_{s,m,d}-1} su_{s,m,d,(t-tt)} = y_{s,m,d,t} \qquad \forall\, s,m,d,t \mid (t-tt) \geq 0$$

$$sd_{s,m,d,t} = su_{s,m,d,(t-UT_{s,m,d}+1)} \qquad \forall\, s,m,d,t \mid (t-UT_{s,m,d}+1) \geq 0$$

In these equations, *y, su and sd* correspond to the setup (operation) of a rail section, its startup and shutdown, respectively. *UT* is the uptime (or train runtime) through the section. Note that the startup of a section implies the entry of a train into it, whereas its shutdown implies that a train leaves the section.

In single-line rail environments, trains can only cross each other in opposite directions if there is a passing loop that is long enough to accommodate one of the opposing trains. Trains can enter an unoccupied passing loop at any time and remain there for an indefinite amount of time (unless specified otherwise) while other trains cross them on the main line. This FHD behavior is well represented by the inventory units in UOPSS. In this case, inventory capacity constraints represent the largest number of trains that can be in the passing loop at any point in time:

$$InvPL_{pl,m,d,t} = InvPL_{pl,m,d,t-1} + FPLi_{pl,m,d,t} - FPLo_{pl,m,d,t} \qquad \forall pl,m,d,t$$

In this equation, *InvPL, FPLi and FPLo* correspond to the inventory of trains in a passing loop, and the "flow" of trains entering and leaving a passing loop, respectively. There are as many "operation modes" for each train section and passing loop as there are train types in each direction. Therefore, if considering two different train lengths, each rail section and passing loop will have four different "operation modes" (*m*): (1) train1-direction1 (T1D1), (2) train1-direction2 (T1D2), (3) train2-direction1 (T2D1), (4) train2- direction2 (T2D2), as illustrated in Fig. 2.

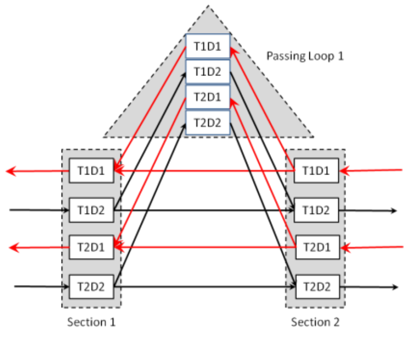

Figure 2. UOPSS representation of single line rail sections and passing loops for 2 train types

Flexibility in the scheduling of rail systems is increased by many long passing loops and large yard capacity at some of the stations, at the expense of inflated capital requirements. Scheduling efforts are also typically simplified in dedicated lines (single purpose, same train priority, homogenous train and passing loop design). However, in systems with limited passing loop infrastructure and mixed (freight/passenger) utilization, more sophisticated modeling must be adopted. Therefore, one crucial aspect of the model is the ability to define the different connectivity between rail sections and passing loops for each train type. Since UOPSS allows the explicit definition of train lengths for each section as different modes of operation, incorporating limited connectivity into the model becomes a trivial task, as demonstrated in Fig. 3 for both a (short) passing loop 1, which only accepts trains of type 1, and a (long) passing loop 2, which accepts both train types.

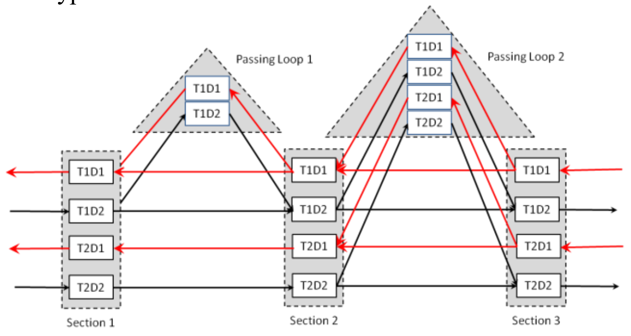

Figure 3. Different passing loop configuration depending on train type

Train travel times (runtimes) through the system are different not only across train types, but also in each direction. This may be due to the existence of hills across the rail system, train load, type of train braking system and train configuration (number and type of locomotives, number and type of wagons). Given the explicit modeling of each train type and direction in UOPSS, it is a straightforward task to assign different runtimes through sections for each train type and direction, which correspond to the "batch times" of individual modes of operation within each section ($UT_{s,m,d}$).

Single- and Double-Line Rail Sections

In limited flexibility environments, single rail tracks (in which only one train can travel in any direction) coexist with double tracks, which accommodate trains in both directions simultaneously.

In UOPSS, the distinction between single- and double line sections is made by the number of terms added to the single-use constraint for each physical section, as indicated by the dotted rectangles in Fig. 4. For single-line sections, all modes of operation (train types and directions) are added to the single-use constraint, whereas for double-line sections, there are two independent single-use constraints: one for each direction. This implies that only one train type can be in any direction in any time period, but two directions may coexist.

Double line sections:

$$\sum_m y_{s,m,d,t} \leq 1 \quad \forall s,d,t$$

Single line sections:

$$\sum_m \sum_d y_{s,m,d,t} \leq 1 \quad \forall t, s = \mathit{single\ line\ section}$$

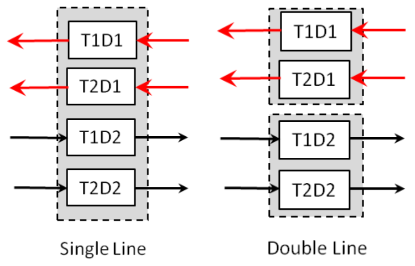

Figure 4. Single- and Double-Line Sections

Trains Following Each Other in Same Direction

Another interesting and complicating characteristic of rail systems is the ability of long trains to follow each other in the same direction. However, when coming in opposite directions, long trains must wait for oncoming trains to completely clear all sections between two long passing loops before it can enter the first section. This behavior is illustrated in Fig. 5.

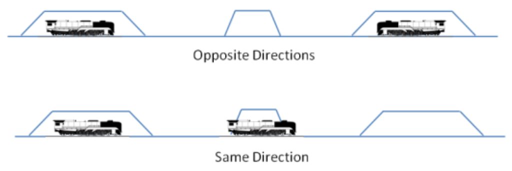

Figure 5. Same- , Opposite Direction Trains

The fact that short trains may use the intermediate (shorter) passing loops and that long trains may follow each other when in the same direction requires extra considerations, given that simply adopting the sections with longer passing loops as a basis for the model would result in overly conservative solutions. This formulation also applies for shorter trains travelling in the same (or opposite) directions in very long single-track sections with no passing loops.

The behavior shown in Fig. 5 is achieved in the model by assigning an extra binary variable for each direction for the long trains that assumes the value of 1.0 whenever any of the intermediate sections is active. A single use constraint on this new binary variable ensures that only one of these larger sections will be active in opposing directions (while still allowing trains to follow each other in the same direction).

$$yL_{ls,m,d,t} \geq y_{s,m,d,t} \qquad \forall ls, s \in ls, d, m, t$$

$$\sum_d yL_{ls,m,d,t} \leq 1 \qquad \forall\, ls, m, t$$

Objective Function

In the previous sections, the UOPSS rail scheduling model structure was described. In rail systems, there are many types of objectives that are of interest to users.

In a recent project, the client-specified rail software was not able to find even a feasible solution for a two day time horizon. This lead to a literature survey of research focused on the train dispatching problem and, to the best of our knowledge, for the complex rail system behavior outlined in the previous sections, no commercial tools exist, nor has previous research been published that addresses all of the previously outlined considerations. Current commercial rail software is often simulation-based, relying on rules and heuristics to avoid and resolve "conflicts" (i.e., infeasibilities) in rail crossings. These tools, even when used by rail experts, are ill-suited to manage the complexity of congested rail systems with multiple train types, single/ double lines and many train stations.

For this reason, the objective of the case studies outlined in this paper was to minimize the usage of passing loops for all train types given a pre-specified train launch schedule where train dispatch times were specified at the battery limits to the problem. This is an indirect representation of the minimization of runtime of trains through the system, which can also easily account for different train priorities if different penalty values are used for passing loop "inventory" variables for each train type. This feature also allows users to set different *spatial* priorities in the usage of passing loops: e.g., passing loop usage closer to a port (or at a crew changeover train station) may be "more acceptable" (i.e., lower penalty value) than delays at other locations.

Another potential objective of this model is maximizing the number of trains running through the system to: (1) determine the rail system's maximum capacity, and (2) identify bottleneck sections. In these cases, the underlying model is identical to the one described in the previous sections, but the objective would be to maximize all of the "train flows" leaving the system in both directions, for all train types. Clearly, these "train flows" should be constrained by the user, lest the optimizer send only the shortest trains through the rail system.

Possible Applications

The first very useful application of this model is to obtain *feasible train schedules* given the train launch timetable. Once again, this far exceeds current commercial rail scheduling software capabilities, which is based on simulation for the detailed rail scheduling decisions and is currently unable to find feasible solutions for a system with this level of complexity.

Since the model is based on a discrete-time scheduling model, management of *track maintenance* becomes a very simple task which only requires setting the specific section's availability (represented by "setup" variables) to zero for the duration of the maintenance operation. The optimization model will find a feasible train schedule that satisfies this requirement, if one exists.

Another interesting and useful application of the model is *recovery from unforeseen events* such as locomotive breakdowns. In this case, after the event is remediated, traffic in the rail system must be re-established as quickly as possible. This is a very challenging requirement for simulation tools, given that many of the rules and heuristics are based on launching trains from stations at the extremes of the rail line, as opposed to from any station in between. For the proposed optimization model, this is a simple task, which only requires forcing setup binary variables to the value of one at start-of-schedule in the sections in which trains are staged.

It is sometimes the case in which rail operators do not know exactly what the *maximum rail capacity* of their system is. Frequently, they rely on historical train schedules and/or hypothetical estimates of maximum capacity. The proposed optimization model allows for a more accurate estimation of the useful rail system capacity, which may then be maximized for the existing track infrastructure as discussed in the previous section.

Another very important application area is the optimal phasing of *expansion of rail infrastructure* by either extending existing passing loops, or adding new ones to the system. This can be achieved by adding a binary variable that corresponds to the usage of each passing loop expansion option, and penalizing it with its corresponding capital expenditure amount in the objective function. Therefore, given the values assigned to each completed train trip, the optimization will only use the extra expansion if the increased capacity benefit outweighs its required capital expenditure.

Results

The detailed train scheduling formulation was applied to a real rail system comprised of 21 sections, 16 passing loops and 5 different train types, representing a mixed-use rail corridor of 370 km. Due to the short travel times between stations, a time step of 10 min was used in the model, and all train runtimes were rounded up to the nearest 10 min to achieve a more conservative solution.

The optimization model was coded in C++ and the problem was solved using CPLEX 12.6 on a laptop with an 8-core Intel Core 2.40 GHz processor and 8GB of RAM. The optimality tolerance for all cases was set to 5%.

Case Studies and Computational Experience

This work started with a 2-day horizon, which was the current best solution achievable with a commercial simulation-based software package. The proposed formulation resulted in a large-scale MILP (Table 1, line 1). Solution statistics are reported in Table 2.

The Gantt chart of a typical 1-week solution is shown in Fig. 6. In this figure, the horizontal axis corresponds to the rail sections, whereas the vertical axis corresponds to time periods (increasing from top to bottom). This "train diagram" is tilted at a 90^{o} angle from its usual format (sections on the vertical axis and time on the horizontal axis) in order to improve its display in the manuscript. Each block color represents one train type, and the columns highlighted in light red represent double rail sections, which may contain trains in both directions simultaneously.

Solve times were quite reasonable for achieving the first integer feasible solution for this large-scale system (about one minute), but CPLEX was taking a longer time (~20 min) to close the integrality gap to within a 5% tolerance.

In recognizing the "staircase" pattern in the solution and the special model structure, a modification to the original UOPSS model was proposed: all dependent binary variables, originally declared as continuous variables, were explicitly declared as binary. While this resulted in many additional binary variables in the model (Table 1, line 2), it also allowed the optimization model to become a *pure integer* programming model (i.e., no continuous variables). This resulted in tremendous improvements in solve time and the ability to obtain the provably optimal solution still within CPLEX heuristics. This effect was most likely due to a combination of efficient CPLEX pre-solve heuristics and cuts, a tight UOPSS model formulation and the favorable, cascading solution characteristics inherent to rail scheduling systems.

This is an interesting result, which further reinforces the well-known facts that (1) the number of binary variables is a poor predictor of MILP performance, and (2) in some instances, it may be beneficial to significantly increase the number of binary variables in the problem. These results also suggest that further research efforts should be directed at developing frameworks for systematically identifying which dependent binary variables in a MILP model should be explicitly declared as binary to improve solution speeds – which can be incorporated as part of commercial MILP solvers.

Given the 18-fold improvement in solve time achieved by formulating this model as a pure IP (in which the final solution was achieved in about a minute), the original problem's time horizon was extended to a full week of train departures.

As can be seen in Table 1, this resulted in a model with over 1.3 million binary variables, 1.6 million equations and over 4 million nonzero elements. This model was solved to provable optimality in less than 20 min, still within the CPLEX heuristics routine.

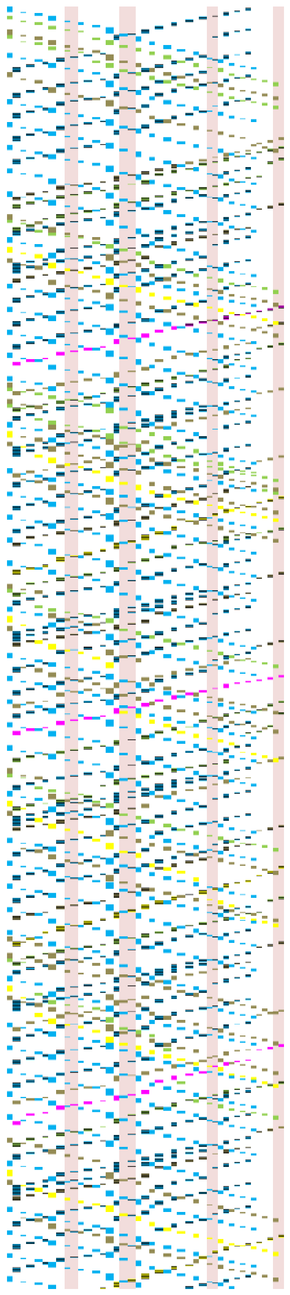

Figure 5. Week-long Schedule, five different train types

Table 1. Model Statistics (Pre-solved values in parentheses)

Case	#Variables	#Equations	#Non-Zeros
2 Days (MILP)	382,474 (11,262 bin, 26,969 cont)	470,994 (63,843)	1,265,737 (214,660)
2 Days (IP)	382,474 (38,328 bin)	470,994 (63,810)	1,265,737 (215,328)
7 Days (IP)	1,338,634 (235,165 bin)	1,627,314 (380,726)	4,412,857 (1,316,240)

Table 2. Solution Performance

Case	1st Feasible Solution (CPU s)	Optimal Solution (CPU s)	Gap at Solution (%)
2 Days (MILP)	74*	1,080 (Node 20,072)	5.0
2 Days (IP)	48*	58*	0.0
7 Days (IP)	1,132*	1,132*	0.0

* Solution obtained within CPLEX heuristics, prior to branch-and-bound

Conclusion

In this paper, a novel, detailed scheduling model for complex rail systems was developed, which far exceeds current commercial tool functionality. The model is based on the UOPSS modeling framework. The proposed approach is very flexible and generic and can be adapted to address any type of rail system.

The proposed model can be used for a number of different purposes: (1) quick determination of feasible train schedules, (2) recovery from unforeseen events in the rail system, (3) track maintenance scheduling, (4) determining maximum rail system capacity, and (5) determining optimal capital expenditures for increased system capacity. This model is the basis of the patent-pending Hatch Rail Optimizer (HRO) software.

The resulting large-scale model was solved for a week-long horizon, and yielded fast solutions using a standard laptop. Significant improvements to solve times were made by converting the original MILP formulation to an IP formulation. From the industrial perspective, this justifies studies on developing frameworks for automatically determining which (dependent binary) variables should be explicitly declared as binary in MILP models to speed up computation.

Nomenclature

$y_{s,m,d,t}$ - setup of section *s* running train *m* in direction *d* in time *t*.

$su_{s,m,d,t}$ - startup of section *s* running train *m* in direction d in time *t*.

$sd_{s,m,d,t}$ - shutdown of section *s* running train *m* in direction *d* in time *t*.

$FPLi_{pl,m,d,t}$ - flow of train *m* into passing loop *pl* in direction *d* in time *t*.

$FPLo_{pl,m,d,t}$ - flow of train *m* out of passing loop *pl* in direction *d* in time *t*.

$InvPL_{pl,m,d,t}$ - inventory of trains *m* in passing loop *pl* in direction *d* in time *t*.

$yL_{ls,m,d,t}$ - setup of large section *ls* with train *m* in direction *d* in time *t*.

References

Andersson, E.V., Peterson, A., Krasemann, J.T. (2015) Improved Railway Timetable Robustness for Reduced Traffic Delays - a MILP approach. *In Proceedings of the 6th International Conference on Railway Operations Modelling and Analysis, Tokyo, 2015*.

Kelly, J.D. (2005) The Unit-Operation-Stock Superstructure (UOSS) and the Quantity-Logic-Quality Paradigm (QLQP) for Production Scheduling in the Process Industries. *In Proceedings of the Multidisciplinary Conference on Scheduling Theory and Applications (MISTA)*. USA, 327-333.

Kondili, E.; Shah, N.; Pantelides, C. C. (1993) Production planning for the rational use of energy in multiproduct continuous plants. *Comput. Chem. Eng*, 17, S123.

Kroon, L, Huisman, D., Abbink,E., Fioole, P.-J., Fischetti, M., Maroti, G., Schrijver, A., Steenbeek, A., Ybema, R.. (2009) The New Dutch Timetable: The OR Revolution. *Interfaces*, 39 (1), 6-17.

Pantelides, C. C. (1994). Unified frameworks for optimal process planning and scheduling. *In Foundations of computer-aided process operations.* NY: CACHE Publications.

Pellegrini, P.,Douchet, G., Marliere, G., Rodriguez, J. (2013) Real-time train routing and scheduling through mixed integer linear programming: Heuristic approach. *In Proceedings of IESM'2013,* Morocco.

Zyngier, D. and Kelly, J.D. (2009) Multiproduct Inventory Logistics Modeling in the Process Industries", in Optimization and Logistics Challenges in the Enterprise. In *Springer Optimization and Its Applications, W. Chaovalitwongse et al. (eds.),* 30, 61-95.