Railway Delay Management:
Exploring Its Algorithmic Complexity

Michael Gatto, Björn Glaus, Riko Jacob, Leon Peeters, and Peter Widmayer

Institute of Theoretical Computer Science, ETH Zurich
{gattom,leon.peeters,rjacob,widmayer}@inf.ethz.ch
glauss@student.ethz.ch

Abstract. We consider delay management in railway systems. Given delayed trains, we want to find a waiting policy for the connecting trains minimizing the weighted total passenger delay. If there is a single delayed train and passengers transfer at most twice along fixed routes, or if the railway network has a tree structure, the problem can be solved by reduction to min-cut problems. For delayed passenger flows on a railway network with a path structure, the problem can be solved to optimality by dynamic programming. If passengers are allowed to adapt their route to the waiting policy, the decision problem is strongly \(\text{NP}\)-complete.

1 Introduction

In recent years, there has been an increasing focus on delays in railway systems, and many railway companies focus on reducing delays in an effort to increase customer satisfaction. The possibility that one train gets delayed is always present. In order to allow passengers to transfer from a delayed feeder train, it can be beneficial to deliberately delay a connecting train. This can save delayed passengers from having to wait for the next train. In this case, some of the feeder’s delay is knocked on to the connecting train, and all passengers already in the connecting train also face a delay. Moreover, these passengers may miss their connection if they wish to transfer in a subsequent station. In general, it might thus be beneficial to propagate the delays in the network, and one should decide on a set of waiting trains. Even today, such waiting policy decisions are usually taken by a human dispatcher. Ideally, a modern railway decision support system should enable a dispatcher to easily evaluate the impact of his decisions, or even propose a good waiting policy. This naturally leads to the algorithmic problem of determining an optimal waiting policy that minimizes the overall passenger delay.

Although delay management problems have been studied for some years, not much is known about the computational complexity of minimizing total passenger delay in general models. In this paper, we analyze restricted variants of the event-activity network model presented in [Nac98], which was analyzed in [Sch02]. In this graph representation, vertices represent arrival and departure events at stations. Directed edges represent driving and waiting activities for...
trains, as well as transfer activities for passengers. Trips of passenger are modeled as paths in the network. Each path has a weight representing its importance. One of the models considered in [Sch02] is based on a bi-criterial objective function. The goal is to simultaneously minimize the perturbation of the timetable and the total weight of paths that miss a connection. This version of the problem is known to be weakly $NP$-hard [Sch02]. The total delay can be efficiently minimized if the “never-meet-property” holds, which basically forbids two delayed vehicles meeting at a station. The general model can be solved to optimality through an ILP-based branch-and-bound algorithm [Sch01,Sch02]. Other theoretical models consider on-line versions with unknown delays at one bus stop [APW02] or the influence of buffer times for delays with exponential distribution for one transfer [Gov98,Gov99]. A fair amount of work was done using simulations [Man01,HHW99,OW99]. However, these last studies are less related to this paper.

As the complexity of the general event-activity network is still unknown, we focus on a restricted setting, as formalized below. The basic element of our model is a direct link between two stations and each such link is operated by a single train. Further, we assume that the original timetable is tight, so there is no possibility for a train to catch up on a delay. In the same spirit we assume that all the transfers at one station happen instantaneously. Hence there is only one amount of delay necessary. Obviously this is a fairly strong restriction of the original model. Nevertheless, we are convinced that the key combinatorial structure remains. As soon as the decision on which passengers to drop are taken, it is easy to produce a modified timetable that minimizes the remaining delays. The general model as well as this model are easily solvable as soon as we know how many passengers use a certain transfer. The problem seems to be that dropping passengers somewhere has a significant effect on these numbers throughout the network. Our model singles out this particular phenomenon. As we cannot even analyze this restricted model in its entirety, it appears that we did not strip away all of the complication of the event-activity network.

For simplicity, we include the externally caused delay of passengers in the objective function. Clearly, such delays cannot be optimized, and their contribution in the objective function is known a priori. As long as we focus on optimal solutions, this offset has no impact on the complexity. Note, however, that this offset can make a considerable difference for approximation ratios and competitive analyses.

We analyze three different cases of our model: (i) a single delayed train in the network with passenger following a predefined path; (ii) origin-destination paths for passenger flows which can have primary unit delays on a railway corridor; (iii) a single delayed train and origin-destination pairs for the passengers, who can adapt their route according to the delays in the network. The primary path delays in the second model may seem unusual, but they should be interpreted as passengers arriving at a station on a delayed train and wishing to transfer. Note that an instance of (ii) can be mapped to an instance of (i) by introducing some additional connections and infinite weight passenger paths. However, this