A comparative evaluation of re-scheduling strategies for train dispatching during disturbances

S. M. Z. Iqbal, H. Grahn & J. T. Krasemann
School of Computing, Blekinge Institute of Technology, Karlskrona, Sweden

Abstract

Railway traffic disturbances occur and train dispatchers make re-scheduling decisions in order to reduce the delays. In order to support the dispatchers, good re-scheduling strategies are required that could reduce the delays. We propose and evaluate re-scheduling strategies based on: (i) earliest start time, (ii) earliest track release time, (iii) smallest buffer time, and (iv) shortest section runtime. A comparative evaluation is done for a busy part of the Swedish railway network. Our results indicate that strategies based on earliest start time and earliest track release time have the best average performance.
Keywords: railway traffic, disturbance management, optimization.

1 Introduction

Today's railway networks are becoming more and more saturated, and even small single disturbances can propagate and have severe consequences. In case of a disturbance, the trains need to be re-scheduled in order to minimize the delays in the railway network. Dispatchers make re-scheduling decision by, e.g., changing tracks, modifying train orders, and changing departure times. The re-scheduling decisions required depend on the disturbance type, and when and where it occurred.

We have addressed the re-scheduling problem by proposing an optimization-based approach [1], implementing a greedy depth-first search branch-and-bound algorithm [2], and implementing a parallel version of this algorithm [3]. These studies have shown that the performance, i.e., the ability to find a good re-


WIT Transactions on The Built Environment, Vol 127, © 2012 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR120481

scheduling solution, is very dependent on which train event that is selected for execution. Therefore, the search algorithm needs good guidance when selecting the most promising candidates to re-scheduling at each step in the algorithm. In this study, we focus on different strategies to guide the search for re-scheduling solutions in the greedy algorithm. We propose and evaluate re-scheduling strategies based on: (i) earliest start time, (ii) earliest track release time, (iii) smallest buffer time, and (iv) shortest section runtime.

The solution quality of the proposed strategies is evaluated using experiments with data from a busy part of the Swedish railway network. The time horizon for re-scheduling is 90 minutes and we study 100 disturbance scenarios in three different categories, i.e., the initial source of delay is (1) a single train is delayed, (2) a single train has a permanent speed reduction, and (3) all trains are delayed on a particular section. All consecutive delays are also correctly modeled.

The results show that the strategies that prioritize earliest start time first and the earliest track release time first have the best average performance. However, we have found that the strategies complement each other, since none of the evaluated strategies is superior in all cases. Further, the strategies are sensitive to how large the initial disturbance is.

2 Related work

Train re-scheduling during railway traffic disturbances is an important problem. An extensive survey is done in [4], where published work is differentiated based on, e.g., infrastructure representation and various traffic properties.

A Mixed-Integer Linear Program (MILP) model is proposed for train re-scheduling of N-tracked railway traffic during disturbances [1]. In [5], the same model as proposed in [1] is used along with two solution methods: (i) right-shift re-scheduling to produce the initial feasible solution and (ii) local search to limit the search.

The train scheduling problem can be formulated as a job shop scheduling problem, as in [6, 7], where train trips are jobs which are scheduled on tracks that are considered as resources. Two studies [6, 8] addressed the problem from the perspectives of capacity, robustness, and dependencies. The heuristics and integer solution methods along with analysis are given in [6]. A variable speed dispatching system is proposed in [7] to control railway traffic by considering acceleration and deceleration time in the model. Furthermore, the work in [7] extends [9] with detailed microscopic and comprehensive models to fulfill additional requirements.

Studies of the computational complexity of disturbance handling in large railway networks are done in [10] and [11]. An experimental study of optimization (i.e. Minimize delay cost) and myopic based policies (i.e. First Come First Served etc.) concludes that both complement each other [10]. The performance of each one is dependent on the region and disturbance type. A performance evaluation of centralized and distributed strategies for dispatching trains is given in [11]. Agreedy depth-first search branch-and-bound algorithm is proposed in [2] to

handle the re-scheduling problem. It generates a feasible solution within 30 sec in most cases. Recently, we have presented a parallel search heuristic [3] based on the greedy algorithm in [2] to reduce the delays.

3 Proposed re-scheduling strategies

The greedy algorithm [2] searches for a solution by trying to schedule all train events in some order, with the goal of minimizing the final delay for all trains at their destination. A train event, e, represents a train movement on a line section or a train stop at a station. In each search step, i.e., scheduling the next event from a candidate list, the algorithm prioritizes the most promising event. Our strategies outlined below select the most promising event based on different objectives.

The greedy algorithm [2] works in three phases when searching for solutions: (i) pre-processing, (ii) depth-first search to find a first feasible solution, and (iii) backtracking and improvement search. The pre-processing phase executes the events that were active at the disturbance time T_0. After that, a lower bound is calculated and a candidate list (NC, sorted according to one of the proposed strategies) is constructed. NC contains the next event to execute for each train. In the second phase, feasible events from the candidate list are executed one by one. When an event has been executed, the candidate list is updated with the next event of the train and re-sorted. The process goes on until a first feasible solution is found. The algorithm then searches for improved solutions in the third phase using backtracking and branch-and-bound until the time limit is reached.

Strategy s_0: s_0 is the strategy implemented in [2] and gives precedence to events that have the earliest start time. It has shown effective to find a first feasible solution. The candidate list, NC, with events is sorted with respect to the following condition: $t_{e'}^{min\_start} < t_{e''}^{min\_start}$, where e' and e'' represent train events in NC and $t_e^{min\_start}$ is the *minimum start time* for an event e. When $t_{e'}^{min\_start} = t_{e''}^{min\_start}$, then $t_{e'}^{min\_start} + t_{e'}^{runtime} < t_{e''}^{min\_start} + t_{e''}^{runtime}$ is used as a second criteria. $t_e^{runtime}$ represents the *section run time* of train event e.

Strategy s_1: The motivation behind s_1, is that strategy s_0 does not consider track release time. A train with long section run time may delay other trains significantly. Strategy s_1 tries to minimize the delay caused by late track release times by sorting NC according to $t_{e'}^{release} < t_{e''}^{release}$. We divide s_1 into two sub-strategies: $s_{1\alpha}$ and $s_{1\beta}$.

Strategy $s_{1\alpha}$ calculates the track release time as $t_e^{release} = t_e^{min\_start} + t_e^{stop} - t_e^{start}$, where t_e^{start} and t_e^{stop} are planned start and stop times, respectively, of event e. If a set of events have the same $t_e^{release}$, we calculate the release time as $t_e^{release} = t_e^{min\_start} + t_e^{runtime}$ as a second sorting criteria.

A railway network has both *station* and *line* sections. Therefore, we introduce strategy $s_{1\beta}$. The track release time is different for events on a *station* as compared to on a *line* section. If an event has a planned stop at a station, then we consider its t_e^{stop} otherwise $t_e^{runtime}$. For both types of sections, the release time is calculated

by condition (1), where s_e is the section type for event e.

$$t_e^{release} = \begin{cases} t_e^{stop} & \text{if } s_e = station \text{ and } planned\_stop = true \\ & \quad \text{and } t_e^{deviation} < t_e^{buffer} \\ t_e^{min\_start} + t_e^{runtime} & \text{otherwise} \end{cases} \quad (1)$$

Strategy s_2: A timetable is designed with buffer times to absorb minor delays, where $t_e^{buffer} = t_e^{stop} - t_e^{start} - t_e^{runtime}$. Strategy s_2 seeks to take advantage of the buffer times and also tries to ensure that the buffer times are fully utilized, thereby aiming at minimizing the delay. The comparison between two events is done based on the condition: $t_{e'}^{min\_start} + t_{e'}^{buffer} < t_{e''}^{min\_start} + t_{e''}^{buffer}$. It behaves as strategy s_0 for events with no buffer time.

Strategy s_3: The strategy s_0 uses the $t_e^{runtime}$ partially when two events have the same start time. Therefore, we introduce a strategy that is based on the *minimum section runtime* (i.e. the minimum time required by each train to use the section) and it is expected to perform well to minimize the delay. It gives precedence to event e' over event e'' as follows: $t_{e'}^{min\_start} + t_{e'}^{runtime} < t_{e''}^{min\_start} + t_{e''}^{runtime}$. Strategy s_3 behaves similarly to s_0 for larger delays.

4 Experimental methodology

In our experimental evaluation we will address how well the proposed strategies perform regarding re-scheduling the trains. The following main questions have been addressed: (i) For how many disturbance scenarios do the re-scheduling strategies find solutions? (ii) How well do the strategies perform relative to each other in terms of total delay at the final destination for all trains? (iii) How sensitive are the strategies to delay variations and the complexity of the disturbance scenarios?

4.1 Input data and setup

We have considered a dense traffic area of Sweden as shown in Figure 1. The 28 stations have 2 to 14 tracks each except Norsholm with only one track. All stations are modeled in detail, including forbidden paths into and out of the stations, see Appendix B in [2].

We use 20 disturbance scenario sets divided into three categories in our evaluation, where each set has five delay variations. In total, we use 100 disturbance scenarios. The sets of disturbance scenarios are described in Table 1. Category 1 has six sets of scenarios where a train has an initial temporary single source of delay, e.g., a train suffers from a temporary delay at one section within the district. Category 2 has seven sets of scenarios where a train has a permanent malfunction resulting in increased running times on all line sections it is planned to occupy. Category 3 has seven sets of scenarios where the disturbance is an infrastructure failure causing, e.g., a speed reduction on a certain section which

Figure 1: The traffic area in Sweden that are used in the study. it has in total 28 stations, all line sections are bi-directional, wide lines indicate double-tracked sections, and thin lines single-tracked.

results in increased running times for all trains running through that section. In all scenarios and experiments we correctly model all consecutive delays that occur.

The greedy algorithm with the evaluated strategies is implemented in Java with JDK 1.6. All experiments are conducted on a server running Ubuntu 10.04 and equipped with two quad-core processors (Intel Xeon E5335, 2.0 GHz) and 16 GB main memory. We have set an execution time limit of 30 seconds for the algorithm.

4.2 Performance metrics

The main performance metric is the total final delay for all trains at their destination. The performance of each strategy, i.e., $s_{1\alpha}$, $s_{1\beta}$, s_2, and s_3, is compared to the base strategy s_0 [2]. We start by comparing for how many scenarios do the different strategies find any solutions, and if their best solutions are better or worse than the best solution for s_0. We measure the relative performance $RP_i^{s_x}$ of each strategy s_x for each scenario i, see Eqn (2), where $C_i^{s_0}$ and $C_i^{s_x}$ denote the final solution value (i.e., the total delay at destination) found by strategy s_0 and s_x, respectively. $RP_i^{s_x} > 1$ means that strategy s_x is better than s_0, and $RP_i^{s_x} < 1$ means that strategy s_x is worse than s_0 for scenario i.

We also calculate the relative improvement or degradation $\mu_i^{s_x}$ for each strategy s_x compared to the base strategy s_0 for scenario set S_i in percentage, see Eqn (2). Our objective is to evaluate the average relative improvement or degradation $\bar{\mu}^{s_x}$ for each strategy s_x. The average is taken over all scenarios, N, where solutions

Table 1: Description of the disturbance scenarios divided into three categories. Each set of scenarios has five different delay variations. These initial delays generate consecutive delays that we also model.

Set	Description
category 1 - Delay variation 6, 9, 12, 15, and 25 minutes initial delay for a specific train	
S_1	Pax train 8762, north-bound, delay Vikingstad-Linköping, having 16 events
S_2	Pax train 538, north-bound, delay Linköping-Linghem, having 20 events
S_3	Pax train 2138, south-bound, delay Katrineholm-Strångsjö, having 9 events
S_4	Pax train 539, south-bound, delay Katrineholm-Strångsjö, having 30 events
S_5	Pax train 8765, south-bound, delay Linköping-Linghem, having 10 events
S_6	Pax train 8764, north-bound, delay Mjölby-Mantorp, having 20 events
category 2 - Delay variation 50%, 75%, 100%, 200%, 300% increased running times for a specific train	
S_7	Pax train 8767 w. permanent speed reduction causing increased run times on line sections starting at Linghem-Gistad, having 13 events
S_8	Pax train 8765 w. permanent speed reduction causing increased run time on line sections starting at Linköping-Linghem, having 10 events
S_9	Pax train 538 w. permanent speed reduction causing increased run times on line sections starting at Linköping-Linghem, having 20 events
S_{10}	Pax train 2138 w. permanent speed reduction causing increased run times on line sections starting at Katrineholm-Strångsjö, having 9 events
S_{11}	Pax train 80866 w. permanent speed reduction causing increased run times on line sections starting at Linköping-Linghem, having 35 events
S_{12}	Pax train 8769 w. permanent speed reduction causing increased run times on line sections starting at Fiskeby-Norrköping, having 20 events
S_{13}	Pax train 539 w. permanent speed reduction causing increased run times on line sections starting at Katrineholm-Strångsjö, having 30 events
category 3 - Delay variation 7, 11, 15, 20, and 28 minutes running time for all trains on a particular section	
S_{14}	Speed reduction for all trains (i.e. 16) between Linghem-Gistad starting w. train 8767
S_{15}	Speed reduction for all trains (i.e. 17) between Linköping-Linghem starting w. train 80866
S_{16}	Speed reduction for all trains (i.e. 19) between Fiskeby-Norrköping starting w. train 8769
S_{17}	Speed reduction for all trains (i.e. 18) between Åby and Norrköping starting w. train 2138
S_{18}	Speed reduction for all trains (i.e. 14) between Vikingstad and Linköping starting w. train 8762
S_{19}	Speed reduction for all trains (i.e. 14) between Mantorp and Vikingstad starting w. train 8764
S_{20}	Speed reduction for all trains (i.e. 16) between Linköping and Linghem starting w. train 538

are available.

$$RP_i^{s_x} = \frac{C_i^{s0}}{C_i^{s_x}}, \quad \mu_i^{s_x} = \left(\frac{C_i^{s0} - C_i^{s_x}}{C_i^{s0}}\right) \times 100, \quad \bar{\mu}^{s_x} = \frac{\sum_{i=1}^{N} \mu_i^{s_x}}{N} \tag{2}$$

5 Experimental results

5.1 Number of scenarios where solutions are found by the strategies

The overall performance of the different re-scheduling strategies for the three categories of disturbance scenarios is shown in Figure 2. It shows the number of scenarios where solutions are found, and now many of these solutions are better or worse than s_0. The horizontal dotted line shows the number of scenarios (i.e. 83) where the base strategy s_0 found a solution. Strategy s_0 finds a solution for all scenarios of category 1, and it finds a solution in 34 and 19 out of 35 scenarios in category 2 and 3, respectively. The labels on the x-axis represent found (F), improved (I), and degraded (D) solutions, respectively, for each strategy.

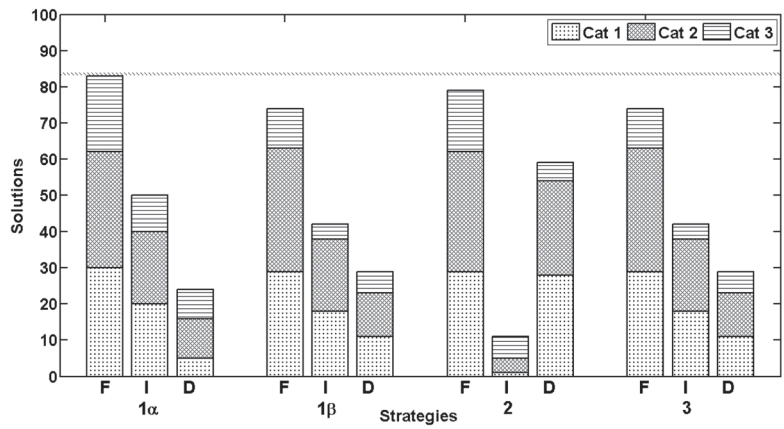

Figure 2: Number of scenarios for with solutions are found for each strategy and for each disturbance category.

As Figure 2 shows, strategy $s_{1\alpha}$ is generally better in terms of number of scenarios where improved and degraded solutions are found in each category. The number of improvements is large for disturbance scenarios of category 1 and category 2 because these are less complex and hence, less risk of deadlock. However, strategy $s_{1\alpha}$ finds only few improvements (i.e. 10 of 21 found solutions) for category 3 scenarios, since they are more complex where all trains on a particular section are delayed.

The performance of $s_{1\beta}$ and s_3 does not differ for any category of scenarios as shown in Figure 2. Strategy $s_{1\beta}$ behaves like s_3 when the event deviation is larger than the buffer time. Strategy s_2 finds improved solutions in only a few scenarios

(i.e. 11 of 79 found solutions) because it, e.g., prioritizes events that have small buffer times but large occupation times.

The strategies are not able to find any solution on average for 23 scenarios out of 100, and most of them belong to category 3. The algorithm goes into a cycle, which means that there is no event in the candidate list that is suitable to execute.

The results show that $s_{1\alpha}$ finds 50 improved and 24 degraded solutions of 83 found solutions. Hence, we conclude that the performance of $s_{1\alpha}$ in terms of the number of improvements is better as compared to the other strategies.

5.2 Relative improvement or degradation

We want to analyze the performance of strategies in terms of $\bar{\mu}^{s_x}$, i.e., average relative improvement or degradation. The value of $\bar{\mu}^{s_x}$ is given in Table 2 for each strategy s_x. The columns show both $\bar{\mu}^{s_x}$ for all scenarios and for the scenarios in each category. For each scenario, two strategies are comparable if both find a solution.

Table 2: Average relative improvement or degradation in percent, $\bar{\mu}^{s_x}$, over the base strategy s_0 for scenarios where solutions are found.

Strategy	Overall	category 1	category 2	category 3
$s_{1\alpha}$	8%	8%	10.49%	2.06%
$s_{1\beta}$	1.65%	-2.24%	6.28%	-1.94%
s_2	-36.51%	-75.81%	-18.41%	-3.54%
s_3	1.81%	-2.29%	6.67%	-1.94%

Starting with the overall results, we find that strategy $s_{1\alpha}$ has an overall positive effect on $\bar{\mu}^{s_x}$ and is on average 8% better than s_0. We also observe that strategies $s_{1\beta}$ and s_3 are slightly better than s_0. Finally, strategy s_2 behaves significantly worse (-36.51% worse) than s_0. We conclude that track release time seems to be a good criteria to prioritize when selecting train events to schedule.

Comparing how the strategies behave for each category, we find that the strategies perform the best for disturbances of category 2, i.e., when a single train has a permanent malfunction causing it to run on slower speed. Strategies $s_{1\alpha}$, $s_{1\beta}$, and s_3 all have better performance than s_0 for category 2 disturbances. For category 3 disturbances, we find that the different strategies have similar performance on average, just $\pm 4\%$ as compared to s_0. Finally, the results also show that strategy s_2 has much worse performance than the other strategies for disturbance categories 1 and 2, i.e., when a single train causes the initial disturbance.

5.3 Sensitivity analysis

In the previous sections, we have evaluated the overall average performance of the different strategies. In order better understand the strategies' behavior and performance we need to do a more detailed analysis of their sensitivity to disturbance category, scenarios, and delay variation. The relative performance $RP_i^{s_x}$ of the different re-scheduling strategies for each scenario is shown in Figure 3, where the base strategy s_0 is the reference line (i.e., the value 1). The x-axis represents the different scenario sets, $S_1, \ldots, S_{20}$, each with five different delay variations.

For the category 1 scenarios, the benefit of the strategies $s_{1\alpha}$, $s_{1\beta}$, and s_3 over s_0 decreases as the delay variation increases. For example, $RP_i^{s_x} \approx 1.4$ for scenario 1 and then $RP_i^{s_x}$ decreases up to scenario 5. A similar behavior is observed also for scenario sets $S_2 \ldots S_6$. For strategy s_2 we observe that it generally has significantly lower performance the s_0. We also observe that the performance of s_2 becomes more similar to the performance of s_0 when the delay increases. The behavior is explained as follows: s_2 considers buffer times when prioritizing among events. When the initial delay increases so will the consecutive delays. As a result, the buffer times will be utilized fully, and strategy s_2 behaves more as s_0. It is interesting to note that the strategies are more sensitive in scenarios of set S_4 (where the delayed train has 30 events) as compared to S_3 (where the delayed train has 9 events). A train with a large number of events may interact with more trains, which causes more conflicts, and then the probability of finding an improved solution decreases. Based on the above observations, we conclude that the performance of strategies in scenarios of category 1 are sensitive to delay variations and the number of events a delayed train has.

For category 2 scenarios, we observe a similar behavior for scenario sets S_7, S_8, and S_9 as for the category 1 scenarios, i.e., as the delays increase the more similar performance have all the strategies. Further, we observe that $RP_i^{s_x} < 1$ for most of the strategies in set S_{11} (the delayed train has 35 events) and in set S_{13} (the delayed train has 30 events). On the other hand, $RP_i^{s_x} > 1$ is observed in most scenarios in set S_8 (the delayed train has 10 events) and S_9 (the delayed train has 20 events) as compared to S_{11}, and these four sets have initial delays on the same section. This indicates that trains with a large number of events interact with a larger number of other trains, thus increasing the complexity of the scenario. Our observations indicate that the performance of the strategies is sensitive to delay variations as well as to the complexity of the disturbance scenario.

For the scenarios in category 3, solutions are found to only few of the scenarios by any strategy. The reason is that all trains on the disturbed section are delayed with different delay variations which contributes to the scenario complexity. Thus, the algorithm either does not find a solution within the execution time limit or ends up in some deadlock situation that it cannot solve. The solutions found are generally for relatively short delay variations. This indicates that the performance of the strategies is more delay sensitive in the scenarios of category 3 as compared to category 1 and 2.

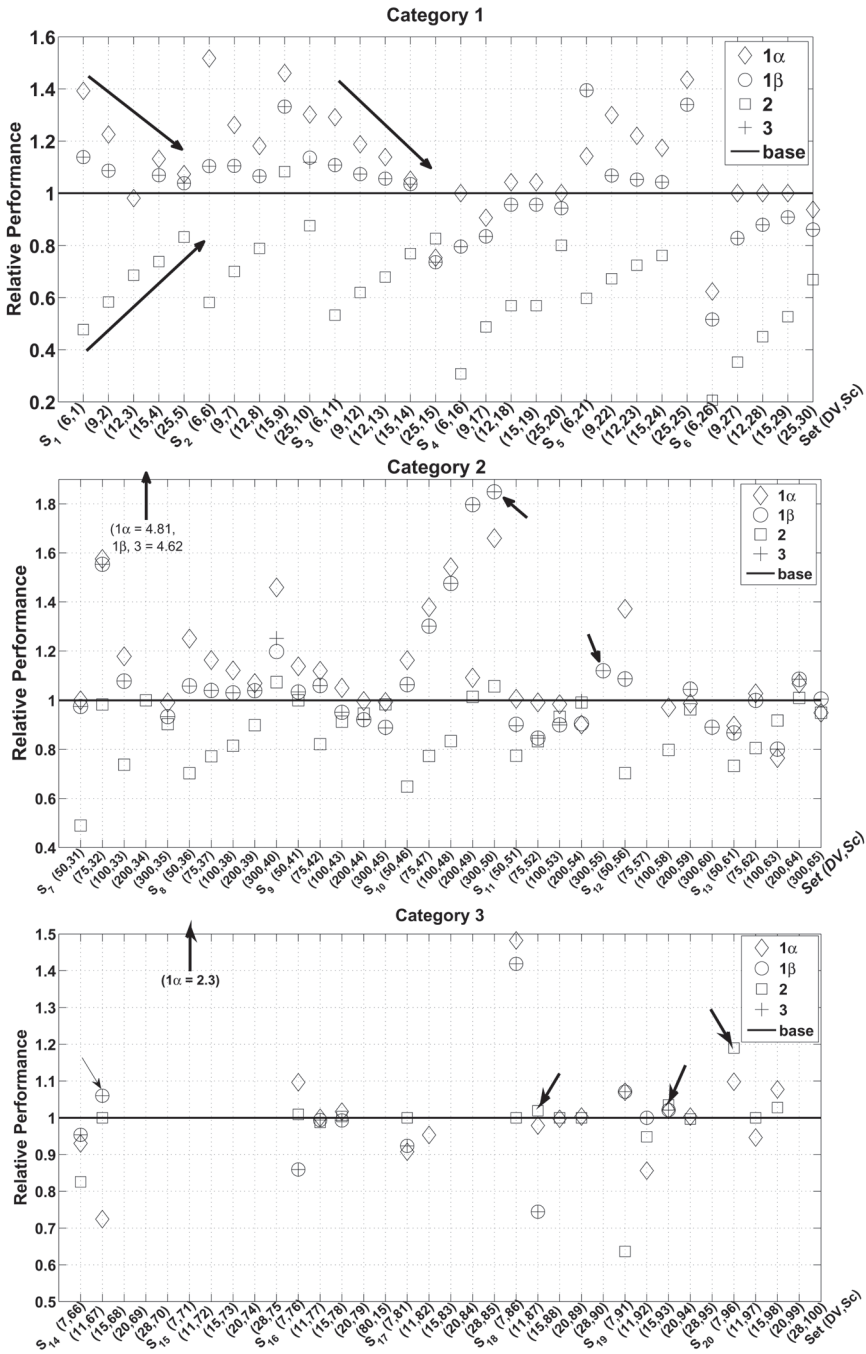

Figure 3: Relative performance, $RP_i^{s_x}$, of the re-scheduling strategies for scenarios of three categories.

Table 3: Selective experimental results for a 90 minutes time horizon and a 30 seconds execution time limit (bold means an improvement over s_0).

Scenario	s_0	s_1		s_2	s_3
		$s_{1\alpha}$	$s_{1\beta}$		
34	20209	**4150**	4376	20209	4376
49	4962	4546	**2762**	4891	**2762**
50	8252	4971	**4462**	7810	**4462**
55	16666	-	**14884**	-	**14884**
67	6627	9151	**6252**	6627	**6252**
71	2772	**1207**	-	-	-
87	4833	4938	6491	**4742**	6491
93	13212	12941	12941	**12765**	12941
96	1310	1193	-	**1101**	-

5.4 Summary and discussion

Some of the found solutions in category 2 and 3 are interesting to analyze more, and they are indicated by arrows in Figure 3. Table 3 lists the solutions found by the different strategies for each selected scenario, where bold means an improvement over the base strategy s_0.

We observe in scenario 34 that strategy $s_{1\alpha}$ significantly reduces the total final delay for all trains as compared to s_0. $s_{1\alpha}$ prioritizes events with early track release times. This is important in order to reduce the consecutive delays for trains with a large number of events (i.e., they may interact with many other trains). Further, on a block based section, i.e., a line section with several consecutive blocks, the scheduling of a train with a large section runtime may increase the consecutive delay for succeeding trains with small section runtimes. Strategy s_3 successfully handles this situation in scenarios 49 and 50. The same solution is found by strategies $s_{1\beta}$ and s_3 in scenario 55 and 67. This is due to the fact that when the event deviation is larger than the buffer time then both strategies behave similarly.

Strategy s_2 reduces the delay in a situation (i.e. in scenario 96) when a train with large buffer times is delayed in favor of trains with small buffer times when their earliest start time is the same. In scenario 71 and 87, the delay on consecutive line sections is reduced. A train must occupy the same track as it used on the previous line section. If the track release time is larger than the time on a second line section then it contributes to the delay. Branching on different tracks may be helpful in delay reduction, and this advantage has been observed in scenario 93.

6 Conclusions

In this study we have evaluated the performance of re-scheduling strategies for train dispatching during railway traffic disturbances. The initial source of delay is of three types: (1) a single train has a temporary delay, (2) a single train has a permanent reduced speed on all line sections, and (3) all trains are delay on a particular section. In total, 100 different disturbance scenarios are evaluated.

Our results show that strategies based on earliest start time and earliest track release time have the best average performance. The earliest track release time strategy has on average 8% better performance than the earliest start time strategy. The strategy based on shortest buffer time has the worst average performance, on average 36% worse than the earliest start time strategy. Our results also show that disturbance scenarios where all trains are affected, e.g., an infrastructure problem on a particular section, are difficult for the strategies to handle. In only 19 cases out of these 35 scenarios were feasible re-scheduling solutions found.

Our variation analysis shows that the different strategies are sensitive to the number of events for a delayed train, i.e., a train with many events may cause consecutive delays for many other trains, and the size of the initial delay, i.e., the longer initial delay the more similar is the performance of the strategies. The quality of solution also depends on scheduling of good candidates on consecutive line sections in scenarios of category 2 and 3 for consecutive delay reduction.

The analysis also shows that no strategy is superior to the others for all scenarios. Therefore, a combined approach can a be a promising alternative. In our future work, we will combine the proposed strategies in the design of parallel algorithms, where different workers can use different re-scheduling strategies when searching for good re-scheduling solutions.

References

[1] Törnquist, J. and Persson, J.A., N-tracked railway traffic re-scheduling during disturbances. *Transportation Research Part B: Methodological*, **41(3)**, pp. 342–362, 2007.

[2] Törnquist Krasemann, J., Design of an effective algorithm for fast response to the re-scheduling of railway traffic during disturbances. *Transportation Research Part C: Emerging Technologies*, **20(1)**, pp. 62–78, 2012.

[3] Iqbal, S.M.Z., Grahn, H. and Törnquist Krasemann, J., A parallel heuristic for fast train dispatching during railway traffic disturbance – early results. *ICORES-2012: 1st Int'l Conf. on Operations Research and Enterprise Systems*, pp. 405–414, 2012.

[4] Törnquist, J., Computer-based decision support for railway traffic scheduling and dispatching: A review of models and algorithms. *5th Workshop on Algorithmic Methods and Models for Optimization of Railways*, 2005.

[5] Acuna-Agost, R., Michelon, P., Feillet, D. and Gueye, S., A mip-based local search method for the railway rescheduling problem. *Networks*, **57(1)**, pp. 69–86, 2011.

[6] Schachtebeck, M., *Delay Management in Public Transportation: Capacities, Robustness, and Integration*. Ph.D. thesis, Niedersächsische Staats-und Universitätsbibliothek Göttingen, Germany, 2009.
[7] D'Ariano, A., *Improving Real-Time Train Dispatching: Models, Algorithms and Applications*. Ph.D. thesis, Technische Universiteit Delft, The Netherlands, 2008.
[8] Conte, C., *Identifying dependencies among delays*. Ph.D. thesis, Niedersächsische Staats-und Universitätsbibliothek Göttingen, Germany, 2008.
[9] Corman, F., *Real-time Railway Traffic Management: Dispatching in complex, large and busy railway networks*. Ph.D. thesis, Technische Universiteit Delft, The Netherlands, 2010. 90-5584-133-1.
[10] Törnquist Krasemann, J., Dynamic railway traffic management during disturbances: Focus on the complexity imposed by deregulation. *ITSWC: Intelligent Transportation System World Congress*, 2008.
[11] Corman, F., D'Ariano, A., Hansen, I., Pacciarelli, D. and Pranzo, M., Dispatching trains during seriously disrupted traffic situations. *2011 IEEE Int'l Conf. on Networking, Sensing and Control (ICNSC)*, pp. 323–328, 2011.