

Improving the quality of railway dispatching tasks via agent-based simulation

C. Biederbick & L. Suhl

*University of Paderborn, Decision Support & OR Lab and
International Graduate School for Dynamic Intelligent Systems*

Abstract

Unavoidable disturbances induce the necessity of operations control and dispatching tasks in the timetable-driven rail traffic. Therefore, dispatching becomes one of the most relevant challenges for economic success, since it directly affects the timeliness of passengers, which is a core indicator of customer satisfaction. Thus, the integration of passenger preferences into dispatching strategies is a reasonable extension of conventional dispatching algorithms. In this paper we present an agent-based simulation system for Deutsche Bahn AG modelling the complete German railway network with about 30,000 trains and millions of passengers per day. With this system various dispatching strategies considering passenger information and aiming to reduce passenger waiting times have been tested. We show that even simple heuristics achieve better results than the rule based dispatching strategies currently in use. Further improvements, qualitative as well as quantitative, are induced by an online re-routing algorithm which efficiently computes new routes for passengers affected by missed connections.

1 Customer-oriented dispatching

Unavoidable disturbances induce the necessity of operations control and dispatching tasks in timetable-driven rail traffic. Dispatchers of a railway have to make decisions about changes to the original train schedule often in a matter of minutes.

Traditionally, railways often emphasise timeliness of trains and cost minimisation within the dispatching process. However, this tends to be suboptimal from the customer point of view. We argue that timeliness of



customers is more important for the long-term economic success of a railway than timeliness of trains.

With customer-oriented dispatching we mean dispatching strategies that give customer timeliness a higher priority than train timeliness. Obviously, we first have to determine measures how to judge customer timeliness, which certainly means different things to different people, thus being an extremely multi-faceted and complex aspect.

Mainly, there are two different aspects of customer oriented dispatching: a) regarding wishes of passengers in the dispatching process through usage of passenger information, and b) proactive and individualised information of passengers using modern communication networks and mobile technology like cell phones, personalised digital assistants (PDA) or so called “smartphones” which are widely spread within industrialised countries.

Both aspects can be implemented and established independently, but obviously they are closely related: The more information there is available, the better the quality of dispatching that can be achieved. In this paper, we focus on aspect a), because it seems to us that there is a lot of room to improve dispatching quality even with little information about passengers.

Although customer orientation is getting more and more important in the competitive passenger traffic market, little is known about how to reliably measure customer satisfaction in railways and how to design dispatching strategies that maximise customer satisfaction. There are several established simulation tools available for railway traffic (cf. [12], as an example), however, they are usually technology-oriented aiming at optimal dispatching of trains and other equipment. In this paper, we report on our activities in simulating railway traffic in a complex network, in order to test various dispatching strategies to maximise customer satisfaction. We present decision support tools which can be used to derive and evaluate explicit dispatching strategies that can be taken over by the responsible dispatchers.

With “dispatching strategy” we mean every algorithm capable of calculating dispatching decisions consisting of waiting instructions for single *connecting trains* waiting for late *feeder trains* in case of a *connection conflict*. In other words, a train – the feeder – is delayed by a disturbance and could not reach its scheduled connectors in time. If a connector does not wait for the corresponding feeder, all transit passengers of this specific connection will be delayed, if it waits, all passengers in the connector get into the risk of missing their connections.

The implementation and test described in this paper were carried out with data from Deutsche Bahn AG. The complete German railway network with about 30,000 trains and about five million passengers daily was implemented within our simulator. With this system, we were able to test various dispatching strategies considering passenger information and aiming in reduction of passenger waiting times.

Topics related to railway dispatching and reliability have been discussed in literature during recent years (see [2, 4, 5, 6], for example), however, in this



paper we take a more explicit view considering customer satisfaction as the main goal of disposition activities.

The paper is outlined as follows: In the second section we describe our agent-based simulation test bed for simulating large railway networks. Section 3 deals with dispatching strategies. We describe how different strategy types in terms of run-time and information demand could be inserted into the simulator. The fourth section discusses an important component of our system, an online passenger router.

In section 5, some computational results using a real world schedule of Deutsche Bahn AG with one million passengers are presented. We close with some conclusions drawn from experimentation.

2 Agent-based simulation of large railway networks

A railway as a system consists of autonomous mobile actors, such as passengers, trains, and personnel. The system components act on a stable infrastructure, such as tracks, stations, and so on. A simulator of railway traffic needs to model all relevant groups of actors.

A natural way to describe mobile autonomous components is to use *intelligent software agents*. Software agents are (broadly speaking) computer programs which are able to behave autonomously in a certain sense, interact with other software agents, thus building communities, and move within a digital network [11]. Because there are usually several interacting agents, we often speak about multi-agent systems. It is conceptually appropriate to use software agents to implement a microscopic view of the railway system, because an agent is usually configured to represent a microscopic item like one certain train or station, even one certain customer.

From the computer science point of view, multi-agent technology can be understood as an advancement of object-oriented programming, thus presenting a new programming paradigm. Especially, the concepts of autonomous behaviour and intentionality provide new dimensions to model objects with control functionality; this is not possible with traditional object-oriented programming techniques.

We generally distinguish between two basic variants of architecture models for multi-agent systems: *deliberative* and *reactive* architecture. Agents in the deliberative model behave analogously to expert systems as they are known in artificial intelligence. They possess an explicit symbolic model of their environment, together with logical inference mechanisms in order to be able to derive conclusions and evaluate possible actions. A well-known deliberative agent architecture model is the "Belief, Desire, Intention" model of [1]. Reactive agents react on certain stimuli from their environment with executing specified actions. The easiest ways of implementation are if-then rules, e. g. *if* obstacle ahead *then* go left. Agent goals are not given explicitly, intelligence arises incrementally a non-centralised way, leading to a new modelling paradigm for distributed systems.



In our opinion, the agent concept is very well suited for modelling parallel and distributed simulation systems running on low-cost hardware. Furthermore, the agent concept enables us to integrate (autonomous) real world players, such as customers and dispatchers, trains and stations, and more specialised agents (the “special agents” will be discussed in forthcoming publications), into the simulation, thus supporting customer information and distributed real-time dispatching as well.

Using the agent paradigm, the simulation model was divided into several regions by simply building disjoint subsets of network vertices (e.g. stations) and arcs (tracks) as described in [3,7,10]. Every region contains one dedicated central dispatching agent, each of them assisted by several task agents, e.g., for carrying out more complex strategy calculations or to re-route passengers missing their connections (cf. Section 3). Then, the size of a (virtual) region is only limited by the dispatching productivity, i.e., the number of decisions computed per time unit. Of course, the regions should be chosen in such a way that communication costs are low within the cluster, and that there are enough communication resources available for peak demands (many delays in the network causing much dispatching activity).

To ensure consistency, a central data storage for infrastructure data (especially the network topology) as well as timetable data (static and dynamic, i.e., with delays during simulation) was chosen. The system is based on the general system architecture introduced in [9].

Between regions, only border crossing trains and passengers are exchanged. Furthermore, dispatchers can receive expected delays and passenger data from trains in neighbouring regions by asking their “colleagues”. This becomes necessary when a connector waits for a feeder train still located in a neighbouring region.

In our simulation model, every complex dispatching strategy is implemented as an agent as well, even when it has no typical agent properties (such as proactiveness or mobility), thus enabling us to use the load balancing mechanisms of the agent-based simulation environment.

The dispatching process is modelled as follows:

- Trains send disturbances occurring on their route to a central server. If a dispatcher has to decide, he will ask this server to receive the latest information.
- A departing train asks the responsible dispatcher for permission.
- The dispatcher computes a decision using different strategies (cf. Section 4) and assisting agents, e.g., the passenger router (cf. Section 3). He also determines whether the train has to wait for some other (technical or security) reason, e.g., congested tracks.
- Resulting decisions will be sent to affected entities in the network, both trains and passengers.
- Additionally, every train and station performs “passenger administration”. All passengers are continuously transferred from one administrative unit to the next one during the course of their trip.



Of course, the dispatcher is the core component of this system. From an agent technology point of view he could be regarded as a deliberative information agent. This agent has to watch the state of the network and to act autonomously in case of present or predictable conflicts. Even if no feasible plan (with no missed connections) could be found, the dispatcher can compute new routes for all affected passengers. Therefore, the passenger router, described in the next chapter, may be used.

3 Online re-scheduling of passengers

One very important extension of traditional dispatching is the on-trip information and re-direction of passengers mentioned above. In our model this is done by passenger routing agents, developed jointly with T. Mellouli, J. Goecke, and DB Systems, (system vendor of Deutsche Bahn AG).

The passenger routers carry out the task of computing alternative routes through the dynamic connection network in case of missed connections. Therefore, a router has to update its internal network model whenever a delay occurs.

There are two ways to use passenger routers in customer-oriented dispatching:

- To determine an optimal passenger re-routing strategy in case of a disturbance taking into account several decision alternatives. Thus, we determine in real-time whether it is better to let some passengers in feeder trains miss their connector or to let some passengers in the connector train miss their connections later on. A dispatching strategy might for example minimise the induced total waiting time in both cases.
- To compute re-routing proposals for passengers already affected by missed connections – this might be helpful for passengers standing on a railway station.

If a disturbance is known before a trip starts, such as the case of a major construction site within the route, the router can be used already before a trip starts.

The internal network representation of a passenger router uses topological sorting and a very efficient update routine. This enables us to include individual restrictions during the search for alternative routes, e.g., personalised minimum transit times for passengers, preferred train categories, or minimum number of train changes.

The importance of efficient re-routing is proven by our computational results in Section 5.

4 Dispatching strategies

It is extremely important to use good dispatching strategies, because they obviously have direct impact on customer satisfaction as measured by delays and missed connections.



In order to rate passenger strategies we chose a simple measure which is directly influenced by dispatching decisions: We measured the individual and total passenger waiting time. Although there are many criteria defining customer satisfaction (cf. [8]), we think (passenger) timeliness to be the most important factor.

In order to properly weight waiting times of customers we have to consider the circumstances under which the waiting takes place. Therefore, we define a parameter called *waiting time quality*: Does the customer wait in the train or at the station? If the latter, what type of station is it, i.e., is there any pastime, or is he or she forced to wait at the perhaps cold and windy platform? A second criterion influencing the quality of waiting time is the type of a trip. Waiting a few minutes longer may not be arduous if the customer is on a holiday trip, while it could be crucial for reaching a business appointment, while in both cases longer waiting times lead to disproportionate annoyances. Thus, passengers should not wait “too long” if dispatching can avoid it. This leads to waiting time costs for each passenger, which could be represented by utility functions (cf. [8]). Many other criteria are possible as well. In reality, the final decision has to be made by the (human) dispatcher: the role of the computer-based system is to provide optimal decision support.

In general, we may categorise dispatching strategies according to the amount and quality of input information they require, as well as according to the art and amount of their running time which may be deterministic or stochastic. Usually, the amount or information required is closely related to the expected running time.

In this paper, we only consider relatively simple strategies with deterministic run-times. Some complex methods, e.g., simulation and mathematical programming, will be discussed in forthcoming publications.

Within the study described in the next section we compared the following basic dispatching strategies:

- General regulations (Do not regard any individualities)

Strategy #1. Do not wait at all (except of security reasons).

Strategy #2. Wait until every feeder train has arrived.

- Decisions regarding categories of feeder trains

Strategy #3. Wait t minutes if the feeder train is of higher category than you.

The existing *waiting time rules* of Deutsche Bahn AG are of this type, too: Only if t is exceeded, a conflict arises and a dispatcher has to decide what to do.

- Time table dependent decisions

Strategy #4. Wait maximally as long as no connection on the route is endangered considering the scheduled arrivals of your own connectors.

- Decisions including passenger information

Strategy #5. Wait, if there are “many” changers in specific feeders. This may be compared with the total number of waiting passengers in the connector, e.g., a given rate q is exceeded, where q is (#changers/#waiting passengers). In real world, this information is usually not available. With simulation, however, we are able to show how useful it could be.



Strategy #6. Variation of #5: Regard the *sum of all changers in all* delayed feeders. If q is exceeded, wait for all feeder trains.

Strategy #7. Variation of #5, but without individual routes. A certain rate of the passengers in a feeder is assumed to change, e.g., based on experienced numbers. This strategy is certainly more easily practicable than the previous ones, because the total number of passengers in a train is fairly easy to estimate.

Strategy #8. Variation of #7 similar to #6.

Strategy #9. Extension of #5: All waiting passengers *at the following* stations are taken into account additionally.

Strategy #10. Extension of #6 in the same way #16 extended #12.

Strategy #11. Extension of #7, again not regarding individual information.

Strategy #12. Analogous to extension of #8.

- Combined strategies

Strategy #13. Use of passenger router: Wait, if at least one passenger in the feeder could not reach the target station in a reasonable amount of time.

For every changing passenger alternative routes are computed. If the estimated new arrival time is close enough to the planned arrival time for all passengers, the connector leaves.

Strategy #14. Compute the total consequences of both the decision that the connector waits for the delayed feeder and that it does not wait, using online passenger re-routing. Choose the alternative with less accumulated total waiting time along all routes and all passengers.

In order to be able to construct an exact model we would need the specific route of each individual passenger over one day. Since this information was not available, we decided to generate “artificial passengers” based on known information about average train contacts per day. We believe that the approximation is good enough to judge effects of various dispatching strategies.

5 Computational results

The strategies described above were tested within the simulation environment. The more promising strategies may be suitable for decision support systems for the human dispatcher in real-life situations as well.

Test environment

The time horizon of the simulation is six hours, i.e., about one million passengers and nearly 9000 trains. Within a simulation run, exponentially distributed delays with expected length of 5 minutes are generated for trains which were chosen arbitrarily. Inter-arrival time of delays is exponentially distributed as well. The delays were chosen in such a way that about 10 % of the simulated trains will be delayed. Note that these are only primary delays; secondary delays could easily be induced by dispatching strategies: a train waiting for its feeders is of course delayed, too.



Our experimentation shows that most simulation runs could be executed in less than two hours on a small cluster of three personal computers (ca. 1.4 GHz and 512 MB RAM). When the passenger router is in use, it runs on a dedicated computer. One single dispatching decision can be computed in a few seconds, even when more complex strategies are in use. (Of course, run-time is mainly influenced by the calculations necessary for the used strategy.)

Some strategies demand the specification of additional parameters, especially those strategies that integrate passenger information comparing numbers of passengers in trains etc. All simulations were replicated 5 times.

Comparison of dispatching strategies

First of all we compared strategies without re-scheduling passengers with missed connections, but penalising a missed connection with a high value in our passengers’ utility functions, arguing that a missed connection implies not only delays for affected passengers, but also disproportionately reduces the quality of the product the customer paid for.

The percentage of passengers missing a connection in our model is comparatively small due to the settings of our delay parameters (see Table 1). This is intended, because no valid estimation of this number in reality was available. In this way, we tend to underestimate the real number which was approved by experts of Deutsche Bahn. Therefore, it is unlikely that we overestimate the impact of online passenger re-scheduling (cf. next section).

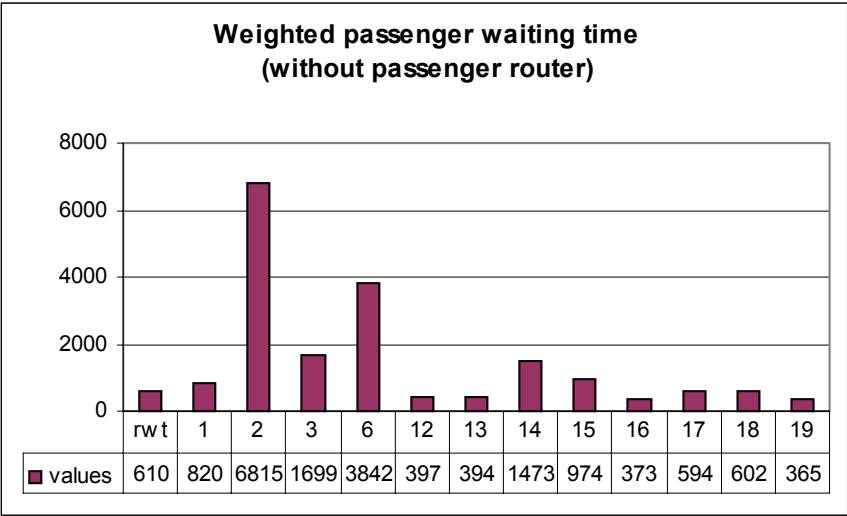


Figure 1: Weighted passenger waiting time of strategies without using the passenger router.



Table 1: Percentage of passengers missing a connection in strategies without the passenger router.

RWZ	1	2	3	4	5	6	7	8	9	10	11	12
0,59	0,79	6,10	1,58	3,43	0,37	0,38	1,35	0,90	0,36	0,57	0,58	0,70

A comparison of the strategies described above clearly proves our hypothesis, that the strategies including passenger information in many cases achieve much better results than waiting time rules of Deutsche Bahn AG do.

Although the regular waiting time (rwt) of Deutsche Bahn AG performs well, there are strategies integrating passenger information which reduce the weighted passenger waiting time down to about 60 %.

Remarkably, our model showed that rwt in general achieves only about 70 % of passengers reaching their destination on time. *Ceteris paribus*, passenger-oriented strategies achieved an average rate of about 90 % (with low variance) in our simulation runs.

Impact of passenger re-routing

Figure 2 demonstrates the effect of using the passenger router (strategies 13 and 14 obviously cannot be simulated without passenger router). It is obvious that strategies utilising passenger information perform much better than others.

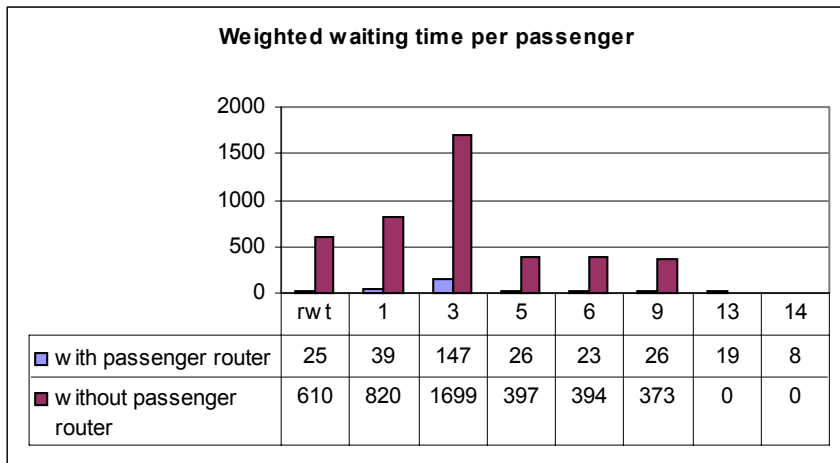


Figure 2: Comparison of weighted passenger waiting time with and without passenger router.

Strategies using the passenger router in advance, i.e., to make a decision, achieve significantly better results than all other strategies, even if the penalty for a missed connection is high in our parameter settings.

6 Conclusions

With this paper we show that agent-based simulation under usage of estimated or exact passenger information could give valuable advice on how to improve the quality of the railway dispatching process.

Furthermore, a customer oriented approach is chosen, emphasising the timeliness of customers in the connection network: some simple dispatching strategies were presented, proving that even the use of estimated passenger data could enhance the process from the customers' point of view.

However, although the simulation model was validated carefully, it is not totally sure that the presented "best" strategies would perform in reality as well as they do in simulation, because all passenger routes are not taken from real-world data, but generated by an algorithm under certain (plausible) assumptions. Therefore, tests with more reliable passenger data would be necessary to evaluate strategies correctly, i.e., we explicitly do not claim to have determined the best possible strategy, but we provide a system which enhances testing of strategy candidates.

As the results in Section 5 prove, availability of early information (and thus re-scheduling of passengers) significantly improves the dispatching results. Therefore, modern proactive information systems pushing individualised information are an important field of further research in order to maintain competitiveness of rail traffic providers.

References

- [1] Bratman M.: Intention, plans and practical reason. Harvard University Press, Cambridge, MA, USA, 1987.
- [2] Carey M.: Ex ante heuristic measures of schedule reliability. *Transportation Research Part B*, 33, 473-494, 1999.
- [3] Goecke J., Biederbick C.: Simulation in Konfliktmanagement in großen Transport-Netzwerken. In: Inderfurth et al. (Hrsg.): *Operations Research Proceedings 1999*, Springer-Verlag, Berlin et al., 2000.
- [4] Goverde R.M.P.: Synchronization control of scheduled train services to minimize passenger waiting times. 4th TRAIL Year Congress 1998, Conference Proceedings, Part 2, TRAIL Research School, Delft, 1998.
- [5] Martin U.: Die dispositive Lösung von Fahrweg- und Belegungskonflikten. Leipzig Annual Civil Engineering Report No.3, 311-333, 1998.
- [6] Shen S., Wilson N.: An optimal integrated real-time disruption control model for rail transit systems. In: Voß S., Daduna J. (Ed.): *Computer-Aided Transit Scheduling, LNEMS*, Vol. 505, pp. 335-364. Springer, Berlin et al., June 2001.
- [7] Suhl L., Biederbick C.: Verteilte Simulation mit Softwareagenten zur Unterstützung von Dispositionsentscheidungen im schienengebundenen Personenverkehr. In: Panreck K., Dörrscheidt F. (Hrsg.): *Tagungsband*



- ASIM Symposium Simulationstechnik 2001, Fortschrittsberichte Simulation, SCS-Europe, Gent, Belgium, 2001.
- [8] Suhl L., Biederbick C., Klierer N.: Design of customer-oriented dispatching support for railways. In: Voß S., Daduna J. (Ed.): Computer-Aided Transit Scheduling, LNEMS, Vol. 505, pp. 365-386. Springer, Berlin et al., June 2001.
 - [9] Suhl L., Mellouli T. Requirements for, and Design of, an Operations Control System for Railways. In: Wilson N. (Ed.): Computer-Aided Transit Scheduling, LNEMS, Vol. 471, pp. 371-390. Springer, Berlin et al., 1999.
 - [10] Suhl L., Mellouli T., Biederbick C., Goecke J.: Managing and preventing delays in public transportation by simulation and optimization. In: Pursula M., Niittymäki J. (Ed.): Mathematical Methods on Optimization in Transportation Systems. Kluwer Academic Publishers, 2001.
 - [11] Woolridge M., Jenkins N. Intelligent Agents: Theory and practice. The Knowledge Engineering Review, Vol. 12, No. 10, pp. 115-152, 1995.
 - [12] Zhu P., Schnieder E.: Determining Traffic Delays through Simulation. In: Voß S., Daduna J. (Ed.): Computer-Aided Transit Scheduling, LNEMS, Vol. 505, pp. 387-399. Springer, Berlin et al., June 2001.

