Task oriented support for train shunting planning

Wout van Wezel
Faculty of Management and Organization
University of Groningen
Groningen, The Netherlands
w.m.c.van.wezel@rug.nl

Keywords: train shunting, scheduling, human computer interaction

Abstract. Train shunting planning for the Netherlands Railways is a complex planning problem that is performed by about 130 full-time planners. It concerns the planning of day-to-day shunting operations at the large stations in the railway network. In our study, the central question is: how can the planners be supported in their task with an advanced planning system? In contrast with the traditional way to design a scheduling support system, we based our system on an extensive task analysis. We describe the results of the task analysis, the design paradigm of task-oriented scheduling algorithms, and the resulting prototype.

1. Introduction
Train shunting scheduling for the Netherlands Railways is a complex planning problem that is performed by about 130 full-time planners. It concerns the planning of day-to-day shunting operations at the large stations in the railway network. Input for this planning is based on the long term schedule of arrivals and departures at a station, rolling stock and track maintenance schedules, and specific circumstances that have to be taken into account when preparing the plan for a particular day or night. Currently, the planners mainly work manually. The plans are made and revised on paper first, after which the outcome is put in the computer. In our study, the central question is: how can these planners be supported in their task with an advanced planning system?

Advanced planning systems offer many advantages. Graphical manipulation of the plan, real time constraint checking, plan evaluation, real time data exchange, and automatic plan generation are functions that are commonly offered by advanced planning systems. Unfortunately, train shunting planning gets very little attention in literature. From a somewhat abstract perspective, shunting can be depicted as a combination of inventory location planning, routing, and staff scheduling. This, however, does not do justice to the specific aspects of train shunting planning. As a consequence, it is not trivial how these functionalities could be applied for shunting planning. To overcome the lack of knowledge about how to support shunting planning, we apply a task oriented approach to resolve our research question. In a task oriented approach, an analysis of the task as it is performed by the planners is input for the design of the system. In the paper, we describe the results of the task analysis and the developed prototype.

In section 2, we shortly elaborate upon the functionality of advanced planning systems. Section 3 describes the train shunting problem in more detail, and in section 4 we describe the developed prototype. Section 5 provides conclusions and directions for further research.

2. Advanced planning systems
Advanced Planning Systems are tools that help in creating detailed plans and schedules. Such systems are basically like any other kind of decision support system consisting of a database, a user-interface, a model base with algorithms, and links to external systems. The database stores all information, the user-interface shows the stored information and provides possibilities to update and manipulate the information, and the model base contains generation algorithms. Scheduling specific functionalities are a constraint checker, a goal evaluator, and a blackboard where multiple solutions and partial solutions can be stored for later retrieval. Figure 1 shows a typical architecture using these components. Literature contains many descriptions from different perspectives (e.g., empirical approaches, system engineering approaches, cognitive approaches) for the various components.

![Figure 1. Scheduling system architecture](image)

The two parts that interact with the human planner are the user interface and the algorithms.

Literature provides little theory on user interfaces for planning and scheduling. Most advanced planning systems are created for production planning, staff scheduling, or routing. Scheduling specific user interface elements in such systems are for example a Gantt chart, a machine dispatch list, a throughput diagram, a performance evaluation interface, and a resource calendar (Pinedo & Yen, 1997, p. 374). Literature on graphical user interfaces for train shunting planning is not available. For the design of our system (which will be described in section 4), we have therefore taken the current representation used by the planners as a starting point.

In sharp contrast with the amount of literature on user interfaces for scheduling, there is much literature on scheduling algorithms. There are two approaches in the development of such algorithms. In the first approach, the developer of the algorithm describes the characteristics of the scheduling problem in variables in a mathematical model. Then, an algorithm is devised that can solve the model, i.e., the algorithm finds values for the variables such that a good solution for the problem is found. In the second approach, the developer looks at the way in which the planners solve the scheduling problems. In contrast with the first approach, the primary aim is not to create an algorithm that solves as much of the scheduling problem as possible. Instead, algorithms are explicitly designed to setup a collaboration between the planner and the system. This latter approach very well fits with the task oriented approach we applied. In section 4, we describe the algorithms we developed to be used in the system. First, however, the train shunting problem is discussed.
3. Train shunting planning

Trains that arrive at a station do not necessarily leave in the same configuration. A train can be split in its individual coaches and these coaches can be combined again in a train. For example, the train from Groningen to Zwolle consists of two coaches and the train from Leeuwarden to Zwolle consists of one coach. In Zwolle the trains are connected, after which they leave as one train to Den Haag. During the night the passenger trains stay at the station. The ‘storage’ capacity, however, is limited. In addition to changing the configurations during the night, all trains must be washed at a track that contains the washing equipment. The task of the planner is to plan the movements of the trains and coaches and to decide on what track trains can stay. To plan the movements, the planner must assign train drivers, train shunters (the ones who connect and disconnect coaches), and routes to move trains from one track to another. The major bottleneck in the shunting yard that we investigated is the washing track. Since the time it takes to wash a train is largely independent from the size of the train, it is advantageous to combine carriages in trains before they are washed. Therefore, the nightly transition can not be solved in one stage, since trains are shunted during the night for washing as well.

Because the main timetable is stable, planning mainly means adjusting existing plans. Changes are made on the basis of events, for example a track that needs maintenance, an extra train that will pass through the station because of a pop concert, etc. The plans are made manually on paper, after which the result is entered in the computer. Figure 2 shows an example of a part of a shunting plan. The computer can check the validity of the plan and send the data to traffic control.

Approximately 130 planners from five major stations in The Netherlands plan the shunting operations for all stations. The Netherlands Railways is in the process of renewing all logistical computer systems. Because of the specific characteristics of the shunting planning problem, and because of the many planners that would have to use the tool for shunting planning, four research projects were started to test two approaches to shunting planning support. In three of those research projects, the
emphasis was on developing mathematical algorithms from different perspectives. In a fourth research project, the emphasis was on a task analysis of the shunting planners. In a follow-up fifth project, we have combined the mathematical and task oriented approaches by designing algorithms in a task oriented fashion. In the following section, we describe the prototype that was the result of this approach.

4. Prototype description

In section 2, we have described the main characteristics of advanced planning systems. In this section, we will elaborate on the aspects in which the user is involved: the user interface and the algorithms. First, however, we will give a concise overview of the results of the task analysis that provided the input for the development of the system.

4.1 Task analysis

An extensive task analysis with two planners revealed a hierarchical task structure. The activities of the planners contain the following high level subtasks:

1. Determine nightly transitions. Several trains of the same type stay overnight at the station. Coaches of the same type are exchangeable, so the planner must decide what incoming coach will become what outgoing coach.
2. Determine tracks. From the national timetable, the planner knows the amount, arrival/departure times, and tracks of both incoming and outgoing trains. After subtask 1, the planner can search for tracks that the coaches can stay on during the night. Because there are much more coaches than shunting tracks, multiple coaches are put on one track. Therefore, the dominant constraint is that a coach should not be blocked by trains to the left and right of it at the time it must be moved to the departure track.
3. Determine routes. When subtask 2 is finished, the planner must make sure that a movement is indeed possible by determining the route the train should use.
4. Assign shunting staff. The route that is used determines the amount of time that is needed to move the train and the time that is needed for shunting staff to walk from one task to another. Consequentially, assigning tasks to shunting staff is the last planning subtask.

The structure is hierarchical because each decision constrains the lower decisions. For example, the possible routes are determined by the tracks that the train will be on. These tasks are performed in this order no matter the kind of event that triggered the task, but always for a few coaches at the time. In other words, the planner does not perform subtasks sequentially for all trains, but only for a limited set of related trains at a time.

Although the task order is fixed, the kind of event determines what task is started with. For example, when a track is scheduled for maintenance, the planner will start with subtask 2 and try to find new tracks for the trains that are scheduled for the track on which the maintenance will take place. Only if he can not find tracks (subtask 2 is overconstrained), he will change the nightly transitions and from there perform subtasks 2, 3 and 4. The consequence of changing a nightly transition will be that a part of the existing schedule becomes obsolete and that subtasks 2, 3, and 4 have to be done again for much of the trains. This means much work and therefore the planner tries to avoid this.
In the prototype, we provide support for subtasks 2, 3, and 4. The reason for this is that in the task analyses and experiments we have focused on the day-planners, for whom subtask 1 is not a frequently performed task.

4.2 Algorithms
Traditionally, scheduling algorithms tend to neglect the human as a decision maker; literature about human interaction with scheduling algorithms is scarce. Moreover, the literature that pays attention to human/algorithm interaction reasons from the perspective of the algorithm, e.g.,

- The planner can choose from a number of alternative solutions that are generated by algorithms (Lauer et al. 1994; Ulusoy & Özdamar 1996).
- The planner may specify weights on goal functions (Smed et al. 2000; Gabrel & Vanderpooten 2002), after which the algorithm generates a schedule.
- The planner can steer the backtracking process of the algorithm (Bernardo & Lin 1994).
- The planner can specify parameters for the algorithm (Ulusoy & Özdamar 1996; Dockx et al. 1997; Oddi & Cesta 2000; Meyers et al. 2002).

Such algorithms, however, are based on domain analyses. In other words, when creating an algorithm, the developer looks at the characteristics of the “objects” that must be attuned such as machines, jobs, staff, shifts, vehicles, etc. In general, there are three phases in using such algorithms:

1. Translation of the domain to the model. This means a formalization of the objects that must be attuned, including their characteristics, constraints, and goals. Some examples of types of models are Linear Programming (LP) models, Traveling Salesman Problem (TSP) models, Assignment models, and Constraint Satisfaction Programming (CSP) models.
2. Solving the model. A model contains a number of variables, constraints, and goal functions. The variables must be assigned values such that the constraints are not violated and the goal functions get the highest possible values. Often, the kind of model that is chosen goes hand in hand with the solution technique. For example, in order to use the Simplex method, the model must be formulated as a LP model.
3. Translation of the solution to the domain. Once a solution has been found the values of the variables must be translated back to the domain.

The paradigm in algorithm development is often that algorithms can solve scheduling and planning problems better than humans. This is true, but only when we look within step 2. When a model has been formulated, the computer is better and faster at finding the best conflict-free solution or a solution that violates as few constraints as possible. However, we must take into account that (a) information might be lost during the translation, and (b) the solution must be interpreted and understood by the human planner in the context of the actual situation. In other words, the algorithm is only a part of the planning process that is for the rest driven by human decision making.

To overcome these problems, we based the algorithms on the subtasks that are performed instead of on the domain. The design of such algorithms requires a different model-building process. From the field of cognitive science it is known that a decision maker implicitly makes a trade-off between the (cognitive) cost of applying
a decision aid (efforts to understand and employ the model and process the information) and the expected benefits (increased quality and speed of obtaining a solution). Decision aids (such as a planning tool or an algorithm) can be improved by applying the following three steps:

- Decompose the planning problem into sub problems and obtain estimates for the efforts (costs) to manually find solutions to these sub problems.
- Identify the sub problems with a high potential of effort (cost) reduction for the decision maker and identify a decision aid that reduces the total effort to find and use a solution for such a sub problem.
- Incorporate specific features for automating storage, retrieval, and computational tasks in the decision aids to manipulate the cognitive effort associated with using these decision aids (Benbasat & Todd, 1996).

In order to investigate whether this approach will also be worthwhile for scheduling and planning, we have developed a task-oriented scheduling system prototype for the shunting planners at the Netherlands Railways. This prototype implements the idea that algorithms should be created for subcomponents of the task strategy to support the problem solving process. The focus is on the level at which the system and the user communicate (Newell, 1981). In other words, human and system should be problem solving in coincident problem spaces (Prietula et al., 1994, p. 660). In our research, we try to find such coincident problem spaces by looking at the subcomponents of the task strategy of human planners.

Looking at the subtasks on the task structure, we can discern four basic assignment tasks: find a free track or combination of tracks, match incoming to outgoing coaches, route a train, and assign tasks to train drivers/shunters. By implementing algorithms for these basic assignment tasks, all steps in the task structure can be supported algorithmically.

1. Track-finding algorithm. Finding a track for a train that is available during a specific time window by varying the time window and constraints. Criteria that will affect the decision to what track the train should be moved are amongst others: the length of the time interval it can stay at this track, the routing distance (i.e., number of direction changes and total mileage) to this track, the previous activities of driver and/or shunter, and the consequences for future actions with this train (i.e., internal cleaning, external cleaning, routing to the track from which it has to leave in the morning, etcetera). The problem the algorithm has to solve is defined as finding a sequence of partially overlapping time intervals from the moment of the actual move to the moment of departure. Sometimes, an additional feature of the sequence is that the cleaning track must be visited somewhere over time. Therefore, the algorithm will have to include the possibility of stating a set of intermittent nodes (i.e., intervals on the cleaning track) from which at least one has to be included in the final sequence before the departure track is reached. Finally, it has to be possible for the planner to block several tracks that may not be included at all in the final sequence, as they have to be reserved for other purposes such as maintenance or trains running through the station. The track finding problem is solved using a K-shortest path algorithm (Van Wezel & Riezebos, 2005).

2. Train unit matching algorithm. Determine how the coaches that enter the station are matched to the coaches that leave. This is performed by a mixed integer programming algorithm that matches arriving to departing train units. The algorithm is described by Freling et al. (2002). This algorithm can also be used on a high
hierarchical level by deleting large parts of an existing plan and matching train units again.

3. Routing algorithm. Given the current plan and infrastructure information, the inputs for this subtask are the source and destination tracks for a train. The output of the subtask is a list with the shortest feasible routes for shunting the train to the proposed track. A modified version of the undirected K-shortest path algorithm of Shier (1976) is used to determine the K shortest paths from the source track to the destination track. Modification of the algorithm of Shier was necessary in order to determine the occurrence of direction changes in a route, which is the primary optimization criterion in the weighing of routing alternatives.

4. Driver/shunter assignment. The input for this task is the moment of train movements. In Zwolle, there are at night 6 train drivers available. After a movement, the train driver must walk to the track where he must move the next train. The main criterion is minimizing the overall walking distance, but a trade-off must be made with the time that drivers must wait at a track (which means a walking distance of zero) for the next train. Waiting too long means they would go to the canteen, which increases the walking distance. A shortest augmenting path algorithm is used that combines shunting activities in cycles in such a way that within a cycle the walking distance is minimized (Jonker & Volgenant, 1987). Multiple cycles are assigned to a shift. Longer cycles mean less overall walking distance but at the same time longer waiting times. The balance is found by calculating multiple solutions with different cycle lengths and combining the cycles to shifts such that waiting time is minimized without violating constraints.

As mentioned, the algorithms are made for the basic assignment subtasks. This means that there is not an algorithm that creates the whole plan. Rather, the plan is created by making a number of consecutive planning decisions where each decision is either taken by the human or by the algorithm. This allows the planning process to be highly interactive, because the human planner and the algorithm can both participate in each step of the problem solving process. For this, however, the algorithms must be integrated in the user interface. In the next subsection we explain how this was accomplished for one of the algorithms.

4.3 Graphical User Interface
In section 2, we described generic functionality of advanced planning systems. We stated that there is not much literature on user interfaces for planning systems in general, and no literature at all on user interfaces for shunting planning. For that reason, we have taken the paper versions of the plans as a starting point, and added interactivity. For our purpose, the following requirements had to be met:

- The planning screens should follow the hierarchical task structure.
- The screens should allow graphical manipulation (dragging and dropping).
- Screens should be linked real-time. For example, if the planner changes the departure track of a train in one screen, all other screens showing the departure track should be updated immediately.
- The input for and outcome of the algorithms must be highly transparent.

Figure 3, figure 4, and figure 5 show examples of the screens in the prototype that conform to these requirements. Figure 3 shows the main planning screen which is the graphical representation of the paper version shown in figure 2. All kinds of manipulations are possible, for example, moving trains in time, changing the track,
splitting a train in individual coaches, etc. Constraint violations are shown by colors in the bars, and the movements of individual coaches can be tracked easily.

Figure 3 gives a time/place representation of the plan. In figure 4, the shunting yard is shown. It shows for a specific time what coach is on what track. In this screen, the tracks the coaches are on can be changed by dragging and dropping. Additionally, the route of the selected train is shown; blue colored tracks show the route from the previous track to the current track.

In figure 4 and figure 5, an example of one of the interactive algorithms is shown. If a train is dragged to another track, the routing algorithm automatically determines a new route. If the planner does not want the default route, he can choose one of the alternatives as shown in figure 5. Furthermore, the planner can determine the route himself by clicking on the tracks in figure 4. Each time he clicks on a track, that track is added to the ‘via’-list, meaning that the routing algorithm will include that track in the route. Immediately after adding a track to the ‘via’-list, a new route is calculated and shown in the user interface.
5. Conclusions

Once, there might have been the belief that computers and clever algorithms would soon render human planners obsolete. This belief is obsolete itself nowadays, as it is widely recognized that both human planners and computer systems have their own distinctive role in planning. We have tried to contribute to this debate by combining methods from different scientific fields in the design of a shunting planning support system.

The goal of the shunting planning project was to create task support, including algorithms, for shunting planners. In-depth task analyses with planners resulted in a number of task structures. By using these task structures to design planning support we combined cognitive science (for the task analysis) and operations research (for algorithms) to design a prototype planning system. The bottom-up approach of algorithmic design provides an extension to the mixed initiative scheduling approaches found in literature. By describing the creation of the plan as a hierarchic problem solving process, we place the discussion about human versus computer at the level where it belongs: an algorithm should not be created for a planning problem, but for a planner’s subtask. Because of the bottom-up approach that was taken, the algorithms are very flexible in their use.

After the algorithms were implemented and integrated in the prototype, the efficiency of the algorithms could be evaluated by comparing the time it takes to solve problems of several levels of complexity. However, the concept of planner-oriented design of algorithms is much more difficult to evaluate. Whether the algorithms effectively support the task of the planner is currently being researched by empirically testing several kinds of support with a large number of shunting planners.

Acknowledgements

This study has been supported by the Netherlands Railways. We gratefully acknowledge the management and planners of this company. Specifically, we want to thank Dr. L. Kroon and the planners of the Netherlands Railways, location Zwolle, for their willingness to co-operate.

References


