

Incident Management in Transport Planning

TRAIL Research School, Delft, November 2002

Authors

L.D. Aronson, R.P.J. van der Krogt, J. Zutt

Faculty of Information Technology and Systems, Delft University of Technology

© 2002 by L.D. Aronson, R.P.J. van der Krogt, J. Zutt, and TRAIL Research School

Contents

Abstract

1	Introduction	1
2	Example applications	2
2.1	AGV terminal operation	2
2.2	Pre- and end-haulage	2
3	Incident management model	3
3.1	Incident management on the tactical level	3
3.2	Incident management on the operational level	4
4	Development of a planning architecture	5
4.1	Tactical Planners	6
4.1.1	Planning and scheduling	6
4.1.2	Replanning	8
4.1.3	Cooperation	8
4.2	Operational Planners	9
4.2.1	Routing algorithm	10
4.2.2	Incident Management	11
4.2.3	Conflict handling	11
4.2.4	Example	12
5	Perspectives	13
5.1	The Tactical Planner	13
5.2	Operational planner	13
6	Conclusions	14

Abstract

In this paper we introduce an agent-based framework which can be used in a dynamic transport planning environment. Incidents are managed at two levels: a tactical level and an operational level. Agents generating multi-vehicle plans belong to the tactical level, whereas agents operating vehicles belong to the operational level. Agents at both levels are equipped with incident management techniques to deal with incidents occurring at their level. The purpose is to create a transport planning system which is both robust and efficient.

Keywords

multi-agent planning, incident management, transport planning, replanning, artificial intelligence

1 Introduction

The topic of our research is the development and implementation of robust multi-agent planning systems in transportation. Multi-agent transportation planning refers to planning of complex transportation problems that require the cooperation of several semi-autonomous actors to complete them. In the past, special attention has been paid to interorganisational coordination in multi-agent systems and some cooperative planning algorithms have been developed and implemented (de Weerd et al., 2002; Tonino et al., 2002). These algorithms take the plans of a number of agents (or organisations) and offer the possibility, via negotiation protocols, to construct a better joint plan. These algorithms however, do not take into account changes of the original plans of the agents and their effect on the cooperation.

The purpose of this paper is to present a framework where incident management techniques are included in multi-agent planning systems in order to develop robust and efficiently cooperative planning methods. Also, we will present some of the techniques we are using to tackle problems occurring within the framework.

We start with providing some prototypical transportation problems requiring the cooperation of several agents where also incidents are prone to happen frequently. Then we sketch a model of a layered multi-agent planning system capable of dealing with various types of incidents. In this model we distinguish incidents that have to be dealt with at the separate levels. Next, we present a general architecture which can be used as a design for a robust multi-agent transportation planning system. We will elaborate on the separate levels in the architecture and techniques to deal with two typical problems, being the generation of robust plans and path finding, will be presented. Finally, we sketch some extensions of the model to more complicated examples of (multi-modal) transportation planning problems.

2 Example applications

In this section we shortly describe two problems which are representative applications for our planning system. These are just an indication; the potential application area is very broad, as the reader can probably imagine.

2.1 AGV terminal operation

In the Netherlands Europe Combined Terminals (ECT) processes around 75% of the containers that arrive at the port of Rotterdam. In 1988 they have started the development of an automated container terminal (Duinkerken and Ottjes, 2000). Cranes lift the containers on or off a boat and autonomous guided vehicles (AGVs) move the containers back and forth between the ships and the stock yard (where containers can be stored). If the containers have to go inland, this occurs by train, freight trucks or short sea shipping. In such an automated terminal costs are reduced in several ways: less employees and training are needed, higher throughput can be achieved and the terminal is more scalable. On the other hand, several extra problems must be overcome. Since there is no human control (only technical engineers), the terminal must be able to detect malfunctioning AGVs (diagnosis) and take appropriate actions (e.g. slowing down other vehicles in the area). In an automated container terminal, incident management is indispensable and replanning techniques such as described in this paper are necessary for a satisfactory operation of the terminal. Planning is done a few hours before the ship arrives (cargo is reported around that time). It might happen that several hours later a ship arrives with a higher priority, in which case replanning is desired. This can be quite a computational expensive task, since at each moment there are more than 30.000 containers at the terminal.

2.2 Pre- and end-haulage

Pre- and end-haulage (also called intermodal drayage) is usually the truck movement part in multimodal transport. Trains are used to move freight over long distances, but additional truck transport is needed to obtain door-to-door transport from terminal-to-terminal transport. The pre- and end-haulage often accounts for the major part of the transport costs, so reduction of these costs is necessary to make multimodal transport competitive (Morlok and Spasovic, 1994). Although the subject of planning in pre- and end-haulage has been studied (Spasovic, 1990), we are not aware of results considering real-time dynamic planning. Therefore we are developing a planner (Aronson et al., 2002) for this application.

3 Incident management model

The primary goal of this incident management is robustness. However, when handling incidents, efficiency is a second goal. The term efficiency both applies to computation times and to the costs of executing the generated plans. Clearly, there is a trade-off between the different goals. The kind of incident management we are dealing with pertains to distributed environments with autonomous actors planning and executing transportation orders. An essential characteristic of such applications is that no single set of orders can be carried out by one single agent (actor), but instead requires the cooperation of several agents. Such cooperation might be required not only because more than one agent is needed to handle all the orders on time but also because the execution of one order already requires the cooperation of many actors together.

In this section we indicate how such a multi-agent transportation system can manage incidents. We introduce two levels of incident management: the tactical and the operational level. The tactical level concentrates on cooperation between agents, while the operational level tries to repair the consequences of incidents by itself.

In general, an incident is any event from outside the planning system that cannot be anticipated with certainty. In applications like the ones we consider, one can distinguish three sources of incidents:

- Incidents originating from the *clients providing the orders*, e.g. cancellation, time window changes, and changes of source and/or destination. Since the consequences of changing one order might affect the operations of several actors (due to their interdependencies) these incidents have to be dealt with at the tactical level.
- Incidents from the *infrastructure*, e.g. a road block or a traffic jam. Although, for future planning, the tactical level should usually be aware of these incidents, a short-term solution, such as a detour, should be provided by the operational level.
- Incidents that arise from *vehicles*, e.g. a malfunctional engine, skidding wheels, or inaccurate steering or acceleration. These kinds of incidents are primarily a concern of the operational level. Only when the incident is anticipated to have long-term consequences, like being out-of-order for a substantial time period, the tactical level should be notified so no new assignments are given to the vehicle and, if necessary, current assignments can be reassigned to other vehicles.

3.1 Incident management on the tactical level

At the tactical level, we are concerned with finding plans for multiple agents and keeping those plans in line with reality. This includes dealing with the following problems.

- Order changes: if a customer changes an order, e.g. a time window or a location, then the tactical level should consider what the consequences are under the current plan, and account for them.
- New orders: a new order arriving at the tactical level does not have to be a real problem when there is sufficient time between the arrival and the execution of the

order. However, when the order arrives late, it is likely that all vehicles are already busy. The decision if the execution of some other order should be postponed in favor of the new order should be made on the tactical level.

- Reassigning orders: if, on the operational level, an incident causes a severe delay, it may not only affect the current execution of an order, but also the execution of the next orders in the plan of the vehicle in question. Thus it may be necessary to reassign orders to other vehicles. This is a responsibility of the tactical level.
- Recomputation of plans: if an incident has some middle- or long-term effect, e.g. a severe traffic jam or a vehicle in repair, the current plan should be recomputed on the tactical level to include the effects of the incident, and, if necessary, be adapted to meet the constraints again.

3.2 Incident management on the operational level

Operational planners have the responsibility of performing the tasks they have been assigned. This implies that the following problems have to be dealt with.

- Task changes: whenever the tactical level decides to change a task which had been assigned to the operational planner, the route may need to be recomputed to account for the changes.
- New tasks: any new task assigned received by the operational planner needs a route computation and possibly a route reservation.
- Route adaptation: if any incidents occur which trouble the execution of the assigned tasks, the operational planner must try to find a solution such that no constraints are violated.
- Detection and notification of failure: if any task is not carried out within the specified constraints, or it is anticipated that this will happen, the tactical level needs to be notified of the situation.
- Detection and notification of situation changes: if an incident has a long-term effect (such as a vehicle breakdown which needs a repair), the tactical level needs to be aware of the new situation.

4 Development of a planning architecture

We are working on a broad system which can be used to test the combined effects of algorithms on the tactical and the operational level. The testbed should be applied to several situations, both real-life and more theoretical. In this section we discuss the current status of the testbed; the status of the different parts will be discussed separately, as well as the status of the total system.

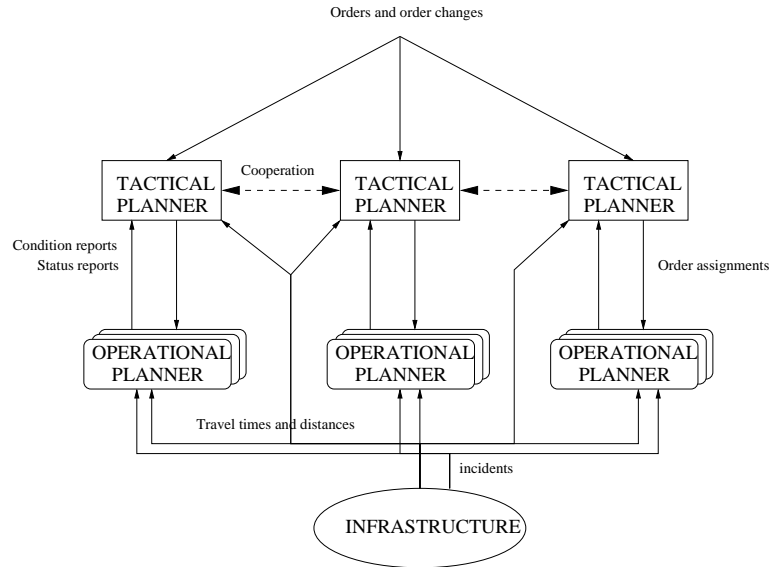


Figure 1: Communication model for planning architecture

Figure 1 depicts the communication model for the total system. The tactical level consists of a number of *tactical planners*. Each tactical planner plans for a set of *transport agents*, which are vehicles with their controlling agents. These controlling agents are called *operational planners*. All planners have access to the infrastructure layout, i.e. all locations, distances, and travel times are known to all planners. This does not necessarily mean that they have all the information at all times, but simply that they can retrieve it when needed. The main input, being the *Orders* or changes in orders, are received by tactical planners. An order contains information about the pickup and delivery locations, the time constraints involved, and the penalty for not executing it within the specified constraints.

Tactical planners consider the available information, including orders, current plan, the cost function, the infrastructure, and incident information, to assign orders to one of their transport agents. Such an *order assignment* contains information about the locations a vehicle has to visit, the time windows in which this has to happen, and which freight has to be transported, if any. An order assignment is sent to the operational planner responsible for the transport agent to which the order has been assigned. Alternatively, a tactical planner may outsource an order to another tactical planner, e.g. by exchanging (parts of) orders. Both tactical planners have to agree that such an exchange is in their common interest.

The operational planner, when assigned an order, finds a route and handles incidents to allow execution of the order assignment. A *status report* is then sent back to the tactical planner, reporting whether the execution has been successful, and if not, how

much delay there will be. Furthermore, in case of an incident the tactical planner should be aware of, a *condition report* is sent so that tactical planner can take the new situation into account.

This section contains a discussion of the problems encountered at the different levels and of the techniques we are currently using to solve them.

4.1 Tactical Planners

As indicated before, the tactical planners focus on efficient distribution of the orders. That is, they receive a stream of orders and order changes as input, and have to produce a stream of order assignments to the operational planners as output. These order assignments specify which order is to be carried out by which agent and at what time.

We have to deal with the following problems.

- **Planning:** this is the problem of *how* the orders should be executed. This includes, for example, determining whether the order can be carried out by a single agent or requires transshipment and, if so, where it should happen.
- **Scheduling:** this problem deals with *when* and *by whom* an order is carried out. This has, by far, the greatest impact on the efficiency and robustness of a solution.
- **Replanning:** this problem deals with how to *repair* incidents.
- **Cooperation:** the final problem is about *working together* with other planners, either to increase efficiency, or by necessity, in case an order cannot be fulfilled by the vehicles managed by a single tactical planner.

We will now discuss these problems in more detail.

4.1.1 Planning and scheduling

For both planning and scheduling, the concept of *slack* plays an important role. The slack between two consecutive locations is the amount of time that can be wasted during the travel, while still being on time at the next location. For example, if the distance between locations A and B is 3 and an agent should visit A at time 1 and B at time 6, the slack would be 2.¹ The slack of an order o_i in a sequence o_{i-1}, o_i, o_{i+1} is the time that can be lost from the moment of delivering o_{i-1} to the moment of pickup of o_{i+1} . This slack is an important factor in tactical plans: if the average slack between two actions in a plan is small, the plan is efficient, but less robust. Hence, the amount of slack determines the trade-off between efficiency and robustness in plans.

A tactical planner keeps a list of assigned orders for each of the operational agents it controls, sorted by time. Upon receipt of a new order, the planner tries to insert it into one of the lists. If there are multiple lists in which the order can be inserted, the planner will have to make a choice. Since the slack is an important determinant of trade-off in robustness versus efficiency, we distinguish two selection heuristics based on slack:

¹For ease of examples, we will assume in this paper that the speed of vehicles equals one distance unit per time unit. In the system there is no fixed speed, however, and different vehicles can have different speeds.

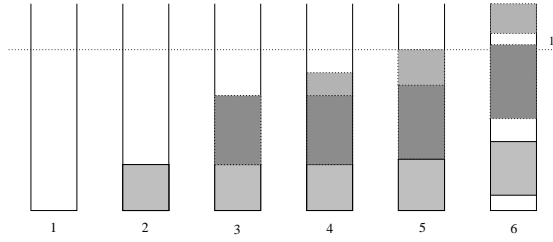


Figure 2: Schematic description of the randomization process

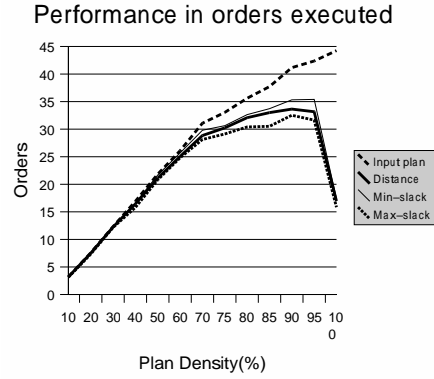


Figure 3: Number of orders successfully scheduled for different plan densities.

slack maximising and *slack minimising*. If it is not possible to insert the order, the planner may try to redistribute the orders, in order to create a plan covering all orders. The rescheduling heuristic first removes all scheduled orders that are not being executed yet and then schedules them again. This time, however, the orders will be sorted first, according to heuristics for the well-known *bin-packing problem*.² With respect to the bin-packing heuristics it is well-known that the order in which the items are examined is important for the quality of the solution. For example, by sorting the items by size in decreasing order, the quality of the solution can be increased from about 70% from optimal to about 22% for a number of well-known heuristics to this problem (e.g. First Fit and Best Fit), (Garey and Johnson, 1979; Krescenzi and Kann, 2000). Similar to these bin-packing heuristics, we will first rearrange the orders and then try to allocate them to the different agents, as if these orders just arrived (all existing orders that are not (about to) being executed are thus removed from the schedules first). The orders will be sorted in decreasing size (determined by sum of the load and unload times and the travel time) and then by priority. Therefore, orders taking a long time to execute will be assigned to agents first, allowing the smaller ones to "fill the gaps". In case not all orders can be assigned, the orders that cannot be fulfilled are examined separately, to see if there is an order with lower priority included in the plan that can be removed to make room for an order not included but with a higher priority. If so, we replace the order with lower priority by the one with the higher priority, thus trying to optimize the schedule for higher priority orders.

Experiments In order to evaluate the different heuristics, we used them to solve a number of planning problems. In these experiments, we did not consider any incidents, and we allowed orders only to be executed in time, or not at all. We measured the distance traveled and the incurred penalties. The setup of the experiments was as follows: first we randomly generated plans for the agents, based on a number of parameters (such as the ranges for load and unload times and penalties, the length of the plans and their

²The bin packing problem is the following problem: given a number of bins of size S and capacity C and a number of items u_1, \dots, u_m of size $s(u) \in \mathbb{Z}^+$, find a distribution of the items over the bins, such that no bin contains more items that it can hold and the least number of bins is used.

density).³ The process of generating plans is visualised in Figure 2. The horizontal line l denotes the desired density level. Steps 1 to 4 show how randomly generated orders are added tightly to the plan, to just below the density level. Then, the load and unload times are randomly increased, until the plan reaches the desired density. Finally, the idle time in the plan is randomly distributed over the plan. This process is repeated for each transport agent.

For each generated plan, we used a slack maximising heuristic, a slack minimising heuristic, and a distance minimising heuristic. The problem instances were generated using the following parameters: agents: 1, 3, or 5; plan densities: 10, 20, . . . , 70, 75, 80, 85, 90, 95, 100%; (un)load times: in the range [0,5]; and penalties in the range [0,10]. We used a fully connected graph of 7 locations, with distances between 3 and 6 as infrastructure. For each set of parameters, we generated 25 instances.

We evaluated the performance of the different heuristics. The results with respect to the number of scheduled orders are shown in Figure 3 (based on 3 transport agents). The figure shows that there is only minimal difference between these heuristics. The runtimes also hardly differ; all problems were solved within seconds. Besides these results we found that, when it comes to measuring the total distance, the distance minimising and slack maximising heuristics outperform the slack minimising heuristic.⁴ In the near future, we will also measure the robustness of the heuristics by considering the sensitivity to incidents. We expect that especially the slack maximising heuristic will perform well with respect to robustness.

4.1.2 Replanning

Several techniques could be used to tackle this problem. For example, general AI replanning strategies, such as the ones we have developed in the past (van der Krogt et al., 2002). These strategies, however, are suited for environments in which complex tasks have to be split into a number of subtasks. As this is not the case in the current testbed, we have opted for a more simple solution. Whenever an incident is reported, the affected orders are removed from the plans. The orders are then presented to the planner again, which reschedules them. Of course, in doing so, the planner has to take into account the reason for the incident, in order to prevent new failures.

Initial experiments have been conducted. However, these experiments have been too small, and too few of them have been performed to report on them here.

4.1.3 Cooperation

Cooperation and coordination are needed because some orders require two or more agents in order to be carried out (multi-modal orders, for example) and also because it may achieve a greater efficiency. We distinguish two basic types of coordination: *cooperative* and *competitive*. In a cooperative environment, the agents are willing to

³The density of a plan is the ratio of the time spent on executing orders versus the total time spent in a plan (including idle time).

⁴Similar results were obtained for larger instances; e.g. the average time for a run with 50 agents and 5000 orders (75% density) is about 48 minutes, in which 96.9% of the orders are successfully scheduled. A less optimal schedule, which schedules approximately the same amount of orders, but accepting a higher total penalty, is obtained in a few minutes.

cooperate if their combined efficiency increases. This is for example the case when we consider different divisions of the same company: one division is willing to accept a small decrease in efficiency if this leads to a bigger increase in the efficiency of the company as a whole. Competitive agents, however, are only willing to cooperate when both agents benefit. This is the case when agents represent different companies. It is clear that cooperative agents may be able to achieve a greater efficiency, since it is less restricted than the competitive setting. In the final system, we aim to include both types of cooperation. Planners can then form groups within which agents work together on a cooperative basis, but also try to cooperate on a competitive basis with agents from outside the groups they participate in.

The current testbed that is used features just one tactical planner. Therefore, we have no technique decided upon yet. Techniques that can be used are *plan merging* (de Weerd et al., 2002) and *distributed plan construction* (de Weerd and van der Krogt, 2002; Durfee and Lesser, 1987). With plan merging techniques each agent first constructs a plan for itself. After that, agents compare their plans to see if there are possibilities for improvements, e.g. combining shipments to improve on transportation costs. The advantage of this technique is that there are efficient algorithms which can be used to compute such improvements. The drawback, however, is that it is not possible to generate the most optimal plans this way, because none of the agents take other agents into account during plan construction. Distributed plan calculation tries to solve this, by letting the agents reason about possible cooperation during plan construction. This allows for more opportunities to exploit possible cooperations. Unfortunately, the only methods that are currently known require the agents to share a great deal of their plans. In a competitive environment, where agents may not wish to share their company secrets, these methods can not be applied. Initial work has begun on methods that require less information sharing (de Weerd and van der Krogt, 2002).

4.2 Operational Planners

At the operational level, planners have to deal with the following three problems: routing, incident management, and conflict handling.

For each incoming transportation order from the tactical planner, the operational planner must compute a route for the transport agent via the pick-up location to the destination location meeting the specified time-windows. We will call this route the initial route for the transport agent.

If there are no unforeseen problems, this initial route will be used to successfully complete the transportation assignment. The most important task of the operational planner, however, is to detect unanticipated situations as early as possible and act appropriately. These situations arise due to the unpredictability of incidents (e.g. vehicle breakdown).

Yet another important problem originates from concurrent planning by multiple transport agents. If, for example, costs of traversing a road depend on the number of transport agents that will traverse the road simultaneously, a planner could compute a transport plan that is in conflict with the plans of one or more other planners. So, another important task for the operational planner is conflict resolution.

4.2.1 Routing algorithm

As pointed out earlier, the operational planner must perform route planning for the transport agents. For this purpose, we have developed the D** algorithm, which is an adaptation of an algorithm for robot's path planning, called the Focused D* algorithm, developed by Stentz (Stentz, 1994, 1995). That algorithm is a dynamic variant of the well-known A* algorithm extended with dynamic arc costs and a focusing function. We will next discuss the development steps from A*, via D* and Focused D*, to D**.

The A* algorithm The A* algorithm is a best-first search algorithm. It keeps track of a list (called the *open* list) of partial solutions to the route finding problem and it tries to expand the most promising alternative first. To determine which alternative is considered best, A* maintains an *estimate* of the final value of each partial solution. This is the sum of the costs of the partial solution and the costs needed to complete the partial solution, the latter estimated by a heuristic function. In order to be applicable the heuristic function must always *underestimate* the costs of the complete solution – containing the partial solution – to produce optimal results (for a proof, see (Russell and Norvig, 1995)). If a heuristic function satisfies this property, it is called *admissible*.

The D* algorithm The D* algorithm is a *dynamic* generalization of A* that takes changing arc costs into account during execution. For example, using D* we can create a traffic-aware routing algorithm, spreading the traffic loads in order to minimize traffic jams (whereas the A* algorithm would have to restart planning from scratch).

Like the A* algorithm, the D* algorithm maintains an *open* list that is sorted by costs. However, states on the D*-list consist of two types: RAISE states and LOWER states. If road costs increase somewhere, a partial path (from the incremented road to the goal) is put on the open list. This path is in RAISE state; if it is taken off the open list, all of its neighbors will be raised likewise (since they end using the same path). After path costs have been raised, it may be better to change the route somewhere (e.g. avoiding a traffic jam). Therefore, RAISE states automatically become LOWER states. LOWER states reduce costs and find the neighbor location with optimal costs.

The Focused D* algorithm The Focused D* algorithm is a variant of the D* algorithm extended with a focusing heuristic. This function directs (focuses) the search toward the goal location. This focusing function is comparable to the heuristic function used in the A* algorithm. A good example of such a function is the function that computes the straight line distance between the current location of the transport agent and the goal location. Obviously, this function is optimistic (underestimating the costs) and therefore admissible.⁵

The D algorithm** Our D** algorithm is an extension of the Focused D* algorithm. The following features have been added:

⁵This statement only holds if all arcs have costs greater than or equal to the air distance between the two endpoints.

- *distribution of information*: transport agents do not have a complete map of the infrastructure, the only requirement is that they can communicate with crosspoint agents at all locations.
- *sensitivity for cost changes*: we only replan if, for a certain road, traversal costs changed significantly. This extension is needed to limit the computation time. In order to use the D* algorithm effectively, one must be able to limit the number of replans.
- *hard time-windows*: the D** algorithm never selects a route that fails to meet time constraints, as opposed to soft time-windows where penalties are assigned for violation of the time-windows. In case there is no possible solution that satisfies all constraints, this is reported to the tactical planner (together with an indication of the seriousness of the problem).
- *alternative routes*: apart from the optimal route we also store several alternatives in order to be able to react quickly to incidents, because in some cases there is no time to compute an alternative route.
- *multi-agent*: the D** algorithm is applied concurrently by multiple operational planners, whereas the Focused D* algorithm was designed for single robot path planning.

4.2.2 Incident Management

In Section 3.2, we listed the different categories of incidents at the operational level. The first category of incidents, originating from the tactical level, change the goals of the transport agent. The transport agent stops whatever it is doing immediately and the D** algorithm is used to plan according to the new situation.

Incidents from the second and third category affect the maximum speed on roads and the driving speed of the transport agent respectively. The D** algorithm can easily recompute local cost changes when at a certain place the maximum speed changes, e.g. due to a road block incident. If the maximum speed of a transport agents changes, e.g. due to problems with the engine, this can be viewed as a series of cost changes of all locations and roads, and can be dealt with by the D** algorithm likewise.

4.2.3 Conflict handling

In general, there are three possible strategies to deal with conflicts: ensure they cannot occur, solve them by negotiating while planning, or solve them by negotiation after planning. A disadvantage of the first option is that a lot of administration is needed: each time an operational planner doubts about traversing a certain road, it must tell the others and check whether other agents doubt about doing the same (and it is not even sure whether this road will really be used in the final planning). A disadvantage of the third option is that if there happens to be a conflict, one or more operational planners have to start (in the worst case, from scratch) over again. If this occurs very frequently (which depends a.o. on the infrastructure and the number of transport agents) it may be worse than the first option. The second option has the disadvantages of both the other options, but to a moderate extend.

4.2.4 Example

To illustrate our operational planner, this section provides two figures created automatically by our operational planner and visualized using the aiSee graph visualization tool (AbsInt, 2002). The graph consists of locations (the boxes) and roads (the edges). The roads are labeled with $[cost : A_1, A_2, \dots, A_n]$, where $cost$ denotes the costs not taking traffic loads into account and A_1, A_2, \dots, A_n are the identifiers of the transport agents that planned to traverse the particular road.

In this example, 10 transport agents (numbered from 0 to 10) all reside in location A. All are directed by the tactical layer to drive to location C. The operational planner uses the D** algorithm to determine a traffic-aware optimal route for the transport agent (in the order of their indices). The result is shown in Figure 4(a). We observe that the first seven transport agents use the route $[A, AR, B, BL, D]$ and the other three the route $[A, AL, B, BL, D]$. The latter have a different route in the beginning, because the previous transport agents crowded the road from A to AR. In the upper part of the figure, between location B and D, this does not occur, because the amount of traffic over the cheapest route does not weigh against the extra costs of the next best route (of course this depends on the capacities of the roads and the weighing constants used in the cost function).

After this planning we introduce a roadblock incident at the road from location AR to location B. The D** algorithm automatically replans for transport agents 0 to 6. As illustrated in Figure 4(b), all transport agents now prefer to travel via location AL instead of location AR.

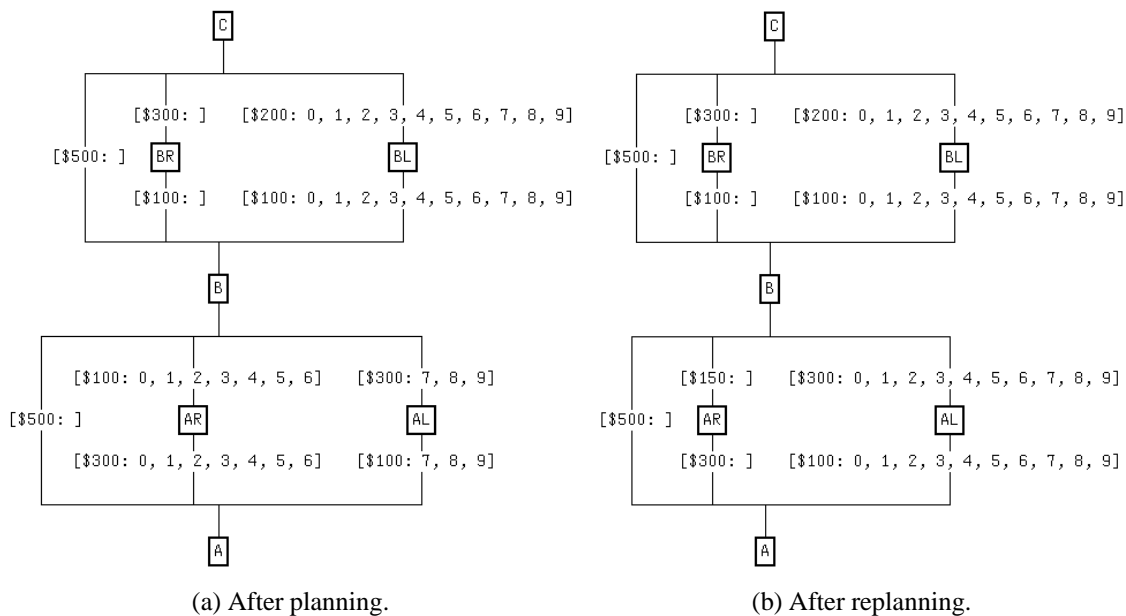


Figure 4: Optional output of the operational planner using Aisee's graph visualization tool.

5 Perspectives

In this section we shortly describe which extensions we intend to do in the near future, at both levels.

5.1 The Tactical Planner

The current allocation policy is simple but inefficient; whenever an order is encountered that cannot be fit into the current plan, the whole schedule is discarded and a new one is made. It would be better to keep the parts of the plan that are efficient, and then reschedule these parts instead of single orders. This has two benefits: firstly, efficient plan parts will not be destroyed and secondly, no computation time will be spent reconstructing these parts. The main drawback is that this heuristic yields locally optimal solutions, which may not be globally optimal. However we suspect that this has not degrade the performance of our planner too much since the heuristic used in the planner already looks for local optimal.

Another area in which we would like to pursue is the multi-agent aspect of the problem. This will allow us to study multi-model transportation problems. Currently, the orders do not have any relationship with other orders and are indivisible. We would like to add constraints to both link orders and break up orders into smaller ones, with the potential of dividing them over multiple agents. In such a situation, the scheduling part could be extended with more general AI planning techniques, as those mentioned in Section 4.1.2.

5.2 Operational planner

The current operational planner does not use all the potential to find intelligent solutions under dynamic circumstances. We need a realistic cost function on the roads, given separation times for the transport agents, which is the minimal distance between two consecutive transport agents. Also, to be able to resolve conflicting operational plans, we need negotiation strategies for operational planners. Finally, to create more flexibility in the operational process, a model extension is needed in which operational planners have more possibilities to change plans on their own. Therefore, we should clarify the border of responsibilities between the tactical and operational planners. At this moment, the tactical planner decides which orders an operational planner has to execute, and also in which order. Since the operational planner has a more detailed view (it only plans for itself, not for any of the other agents) it might be desirable that operational planners cooperate with other agents or change the order of the assignments by the tactical planner.

6 Conclusions

In this paper we have presented a framework for incident management in transport planning systems. The framework consists of a tactical layer and an operational layer. Both layers have their own responsibility when it comes to handling unexpected situations.

The tactical layer distributes orders over vehicles. It therefore has to handle incidents which potentially change the distribution, like the arrival of new orders and the notification of a vehicle that an assigned order cannot be carried out on time. We have developed slack based heuristics. Initial analysis shows that the heuristic is about as good as a distance based heuristic, with respect to the number of orders which is successfully planned and the total distance. Analysis of the robustness has still to be done, but we expect that the slack maximising heuristic will perform well with respect to its sensitivity for incidents.

We are also developing techniques for cooperation between different agents on the tactical level. This has the potential to increase efficiency and robustness. Analysis of these techniques will be done in the future.

The operational layer is concerned with handling incidents influencing the order execution of its assigned orders. This implies routing, incident management, and conflict handling. We have developed a new multi-agent routing algorithm D** which can be used in dynamic situations. Although we have presented ideas to incorporate incident management and conflict handling into our routing algorithm these issues need to be elaborated on in the future.

The main advantages of embedding these layers into a common framework are the fact that both levels are closely tuned and that the total system performance can be measured in addition to the performance of the separate levels. To that purpose we are developing a simulator which can visualize and measure the total system. Especially the resulting robustness will be interesting: to what extent can the system handle incidents without breaking down?

Acknowledgements

The authors are part of the Collective Agent Based Systems (CABS) group at the faculty of Information Technology and Systems (ITS) of the TU Delft. Leon Aronson and Roman van der Krogt are supported by the "Freight Transport Automation and Multimodality (FTAM)" research program. Jonne Zutt is supported by the TNO-TRAIL project "Fault detection and recovery in multi modal traffic networks with autonomous mobile actors". Both programs are carried out within the TRAIL research school for Transport, Infrastructure and Logistics. The projects are supervised by Cees Witteveen.

References

- AbsInt (2002). Angewandte informatik. <http://www.absint.com/aisee/>.
- Aronson, L., Konings, R., and Kreutzberger, E. (2002). Improving pre- and end-haulage in intermodal transport: Development of an efficient planner. In *Proc. of the Int. Cong. on Freight Transport Automation and Multimodality*.
- de Weerdt, M., Bos, A., Tonino, H., and Witteveen, C. (2002). A resource logic for multi-agent plan merging. *To appear in Annals of Mathematics and Artificial Intelligence, special issue on Computational Logic on Multi-Agent Systems*.
- de Weerdt, M. and van der Krogt, R. (2002). A method to integrate planning and coordination. In Brenner, M. and des Jardins, M., editors, *Planning with and for Multi-Agent Systems, Technical Report WS-02-12*, pages 83–88. AAAI Press.
- Duinkerken, M. and Ottjes, J. (2000). A simulation model for automated container terminals. In *Proc. of the Business and Industry Simulation Symposium (ASTC 1999)*.
- Durfee, E. and Lesser, V. (1987). Using partial global plans to coordinate distributed problem solvers. In *Proc. of the 10th Int. Joint Conf. on AI (IJCAI-87)*, pages 875–883. Milan, Italy.
- Garey, M. and Johnson, D. (1979). *Computers and Intractability - A Guide to the Theory of NP-Completeness*. W.H. Freeman and company. ISBN 0-7167-1044-7.
- Krescenzi, P. and Kann, V. (2000). A compendium of np optimization problems. <http://www.nada.kth.se/~viggo/problemlist/compendium.html>.
- Morlok, E. and Spasovic, L. (1994). Redesigning rail-truck intermodal drayage operations for enhanced service and cost performance. *Journal of the Transportation Research Forum*, 34(1):16–31.
- Russell, S. and Norvig, P. (1995). *Artificial Intelligence, a modern approach*. Prentice Hall International Editions, Eaglewood Cliffs, NJ.
- Spasovic, L. (1990). *Planning Intermodal Drayage Network Operations*. Ph.D. thesis, University of Pennsylvania.
- Stentz, A. (1994). Optimal and efficient path planning for partially-known environments. In *Proc. of the Int. Conference on Robotics and Automation*. San Diego.
- Stentz, A. (1995). The focussed d* algorithm for real-time replanning. In *Proc. of the 14th Int. Joint Conference on AI*. Montreal.
- Tonino, H., Bos, A., de Weerdt, M., and Witteveen, C. (2002). Plan coordination by revision in collective agent-based systems. *Artificial Intelligence Journal*.
- van der Krogt, R., Bos, A., and Witteveen, C. (2002). Replanning in a resource-based framework. In Mařík, V., Štěpánková, O., Krautwurmová, H., and Luck, M., editors, *Multi-Agent Systems and Applications II (LNAI Volume 2322)*, pages 148–158. Springer Verlag.