ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008

doi:10.3233/978-1-58603-891-5-929

Robust Reservation-Based Multi-Agent Routing

Adriaan ter Mors and **Xiaoyu Mao**[1] and **Jonne Zutt** and **Cees Witteveen**[2] and **Nico Roos**[3]

1 Problem description

In a multi-agent routing problem, agents must find the shortest-time path from source to destination while avoiding deadlocks and collisions with other agents. Agents travel over an infrastructure of *resources* (such as intersections and road segments between intersections), and each resource r has (i) a capacity $\mathrm{C}(r)$, which is the maximum number of agents that may occupy the resource at the same time, and (ii) a minimum travel time $\mathrm{D}(r)$. Example 1 illustrates this multi-agent routing problem.

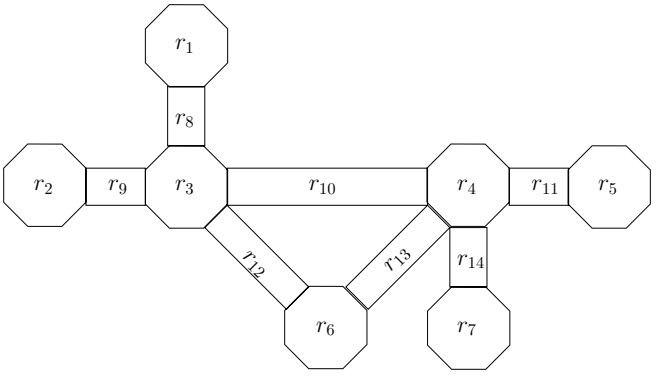

Figure 1: Infrastructure of unit-capacity resources.

Example 1. *Figure 1 shows an example of a multi-agent routing problem. There is an infrastructure of 14 resources: resources r_1 to r_7 represent intersections or interesting locations, whereas resources r_8 to r_{14} represent lanes between the intersections. All resources have a capacity of one and the same minimum travel time. Suppose we have two agents, A_1 that wants to go from r_1 to r_7, and agent A_2 that wants to go from r_5 to r_2. The optimal individual plans for these agents are p_1 and p_2 respectively:*

$$p_1 = (r_1,1),(r_8,2),(r_3,3),(r_{10},4),(r_4,5),(r_{14},6),(r_7,7)$$
$$p_2 = (r_5,1),(r_{11},2),(r_4,3),(r_{10},4),(r_3,5),(r_9,6),(r_2,7)$$

In the above, $(r_1,1)$ in p_1 means that during time unit 1, agent A_1 is travelling on resource r_1. These two plans cannot be both put into action, as they are in conflict with each other: both agents plan to travel on resource r_{10} during time unit 4, but this is not possible, since each resource can hold at most one agent at the same time. There are two ways to solve this conflict. The first is that either A_1 or A_2 (but not both) does not make use of r_{10}, by making a detour along r_6. The second solution is that one of the agents waits until the other has passed. If, as we assume in this paper, we only optimize for time (as opposed to e.g. distance travelled), then the first solution is the best.

The multi-agent routing problem often occurs in application domains of Automated Guided Vehicles (AGVs), such as transportation of goods in warehouses, or loading and unloading of ships at container terminals. The multi-agent routing problem is also relevant in taxiway planning on airports.

The quality of a routing method is judged not only on the basis of its efficiency (i.e., in terms of the time required by the agents to reach their destinations), but also on the basis of its ability to deal with changing circumstances and unexpected *incidents*. Examples of incidents in the application domains mentioned above are human interference with AGVs (e.g. by people stepping in the path of an AGV) in a warehouse scenario, or the delay of a ship (or aircraft) at the (air)port.

2 Reservation-based multi-agent routing

In a reservation-based approach to multi-agent routing, agents plan their route by reserving time intervals on resources; these reservations should be made in such a way that an agent's plan specifies its location (i.e., the resources) at each point in time. Furthermore, the routing method should ensure that reservations of different agents are not in conflict with each other. The definition of a conflict we employ here is simply that the capacity of a resource may not be exceeded at any point in time.[4]

We have developed a routing algorithm that a single agent can use to find a conflict-free plan, given a set of reservations previously made by other agents. Our algorithm is optimal, in the sense that it finds the *shortest-time* conflict-free plan for this agent. The algorithm can be described as a shortest path search through the *free time window graph*, where a free time window is a time interval associated with a resource, in which the resource can accommodate at least one more agent.

Our approach is similar to that of Kim and Tanchoco [1]. However, their distinction between lanes and intersections (as opposed to only having resources), combined with explicitly checking for conflicts results in a computational complexity of $O(\mathcal{A}^4R^2)$, whereas our algorithm has a complexity of $O(\mathcal{A}R\log(\mathcal{A}R)+\mathcal{A}R^2)$. The full algorithm, the proof of its correctness, and the analysis of its worst-case complexity can be found in [4].

[1] Almende BV {adriaan,xiaoyu}@almende.org
[2] Delft University of Technology, {j.zutt,c.witteveen}@tudelft.nl
[3] Maastricht University, roos@micc.unimaas.nl

[4] A more advanced conflict definition, in which agents are not allowed to overtake each other (*catching-up conflicts*) or pass each other by (*head-on conflicts*) can also be modeled in our framework, but we do not show this here.

3 Dealing with incidents

It stands to reason that carefully crafted plans, which detail all actions of all agents at each point in time, may be obsoleted even by minor incidents in the environment. In their survey paper on design and control of AGV systems, Le-Anh and De Koster [2] wrote "*a small change in the schedule may destroy it completely*", referring to the reservation-based routing method of Kim and Tanchoco [1]. The truth of this statement depends on the existence and quality of mechanisms that can *repair* route plans.

The quality of a repair mechanism depends on (i) the cost of the repaired plan in relation to the cost of the original plan, (ii) the similarity between the original and the repaired plan (the more similar the better), and (iii) the computational effort required to perform the repairs. Maza and Castagna [3] proposed a repair mechanism designed to prevent deadlocks that is both computationally inexpensive and adheres closely to the original plan. In Section 4, we investigate the cost of repaired plans when combining Maza's mechanism with our route planning algorithm.

To see how a delay of one agent can create a deadlock, consider again the infrastructure of Figure 1.

Example 2. *We have the same two agents: A_1 with source and destination r_1 and r_7, and A_2 with r_5 and r_2. Suppose they have the following plans, in which A_2 plans to wait (in resource r_{11}) until A_1 has passed:*

$$p_1 = (r_1,1),(r_8,2),(r_3,3),(r_{10},4),(r_4,5),(r_{14},6),(r_7,7)$$
$$p_2 = (r_5,1),(r_{11},2),(r_4,6),(r_{10},7),(r_3,8),(r_9,9),(r_2,10)$$

Suppose that in the execution of his plan, A_1 is delayed in resource r_3 until time 7. To resume his journey, A_1 wants to go to r_{10}, but that resource is occupied by A_2; similarly, A_2 is also stuck, since the next resource in his plan, r_3, is occupied by A_1.

The idea of Maza and Castagna is to determine for each resource which agent will enter the resource first, second, etc. This information can be derived from the plans of the agents. Then, during the execution of the plans, an agent is only allowed to enter a resource when it's his *turn*. In our example, A_2 is the second agent to enter r_4, so it will wait in resource r_{11} until its turn has come, which is after A_1 has exited r_4.

4 Evaluating robustness

We evaluate the ability of our routing method to deal with change (its *robustness*) by measuring the delay caused by the deadlock-prevention mechanism. This *mechanism delay* is the time agents have to wait before they are allowed to enter a resource (or the time they have to wait behind other agents that are waiting for clearance to enter a resource, as we do not allow overtaking in our experiments).

We had the following experimental setup: first, all agents make a route plan for their (randomly chosen) start and destination locations. Then, in a simulation environment, the agents try to execute their plans. If these (reservation-based) plans are executed perfectly, then no conflicts will occur and all agents will arrive at their destination on time. However, we generate random incidents that cause agents to stop for a fixed duration, potentially blocking other agents behind them.

Over different experiment runs, we varied the following parameters: (i) the infrastructure: we used random networks, small-world networks, lattice networks, and a map of an actual airport; (ii) the number of agents in the system; (iii) the frequency and duration of incidents. The frequency is a value p that represents, for every resource in the agent's plan, a chance of p of having an incident.

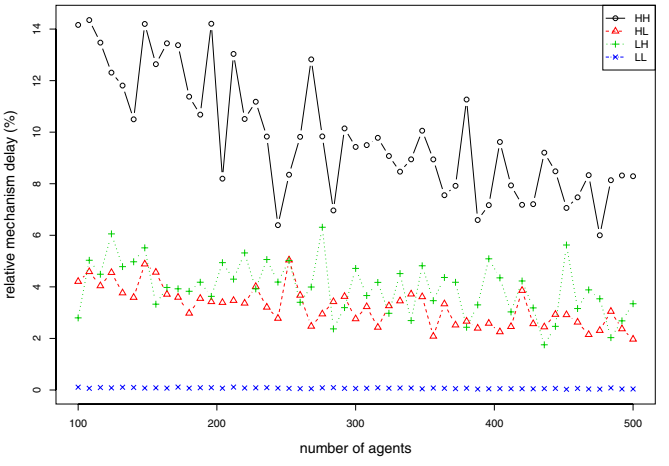

Figure 2: Mechanism delay for Amsterdam Airport Schiphol infrastructure. Incident parameters: HH = (p=0.1, duration=120s); HL = (p=0.1, duration=30s); LH = (p=0.01, duration=120s); LL = (p=0.01, duration=30s).

Figure 2 shows the mechanism delay, averaged over all agents, as a percentage of the plans of the agents. At least three noteworthy conclusions can be drawn from this figure: first, as number of agents increases, the relative mechanism delay decreases. It turns out that the increased congestion in the system is more important that any increase in complexity that might result in more mechanism delay. Second, even for a high number ($p = 0.1$) of long incidents (duration = 120s), the mechanism delay is never more than 15% of planned travel time. Third, for a small number ($p = 0.01$) of short incidents (duration = 30s), there is no discernible impact on plan quality.

Experiments conducted on other types of infrastructures produced figures similar to Figure 2: on lattice networks and small-world networks, mechanism delays were slightly smaller (maximum relative mechanism delays around 10%), whereas for random networks they were higher, with a maximum relative mechanism delay of 30%.

ACKNOWLEDGEMENTS

This research is supported by NWO (Netherlands Organization for Scientific Research), Grant No. CSI4006.

REFERENCES

[1] Chang W. Kim and J.M.A. Tanchoco, 'Conflict-free shortest-time bidirectional AGV routeing', *International Journal of Production Research*, **29**(1), 2377–2391, (1991).

[2] Tuan Le-Anh and M.B.M De Koster, 'A review of design and control of automated guided vehicle systems', *European Journal of Operational Research*, **171**(1), 1–23, (May 2006).

[3] Samia Maza and Pierre Castagna, 'A performance-based structural policy for conflict-free routing of bi-directional automated guided vehicles', *Computers in Industry*, **56**(7), 719–733, (2005).

[4] Adriaan W. ter Mors, Jonne Zutt, and Cees Witteveen, 'Context-aware logistic routing and scheduling', in *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling*, (2007).