

# NO MORE DEADLOCKS – APPLYING THE TIME WINDOW ROUTING METHOD TO SHUTTLE SYSTEMS

Thomas Lienert  
 Johannes Fottner  
 Institute for Materials Handling, Material Flow, Logistics  
 Technical University of Munich  
 Boltzmannstraße 15, 85748 Garching, Germany  
 Email: lienert@fml.mw.tum.de, kontakt@fml.mw.tum.de

## KEYWORDS

Shuttle systems, Routing, Deadlock handling

## ABSTRACT

Autonomous vehicle-based storage and retrieval systems are used in order to supply picking or production areas based on the goods-to-person principle. In one such system, several vehicles move within the same rail system. Hence, routing and deadlock handling is an important issue that has to be resolved carefully to run these systems efficiently and robustly. One possibility for coping with deadlocks is deadlock avoidance by routing with time windows.

In this paper, we present a modelling approach that allows us to apply the time window routing method to shuttle systems. We model the system as a mixed graph and present a concept for moving vehicles safely and efficiently through the storage system.

## INTRODUCTION

In addition to stacker-crane-based automated storage and retrieval systems (AS/RS), a new technology has been introduced to the market, based on autonomous vehicles. Autonomous vehicle-based storage and retrieval systems (AVS/RS) are used for storing unit loads in order to supply picking and production areas based on the goods-to-person principle (VDI- Richtlinie 2692).

AVS/RS, also known as shuttle systems, are characterized by horizontally operating vehicles. These vehicles travel within a rail system that is integrated into the storage rack. Lifts positioned along the periphery of the storage rack system are used to perform storage and retrieval transactions (Malmborg 2002).

Over the course of recent developments, different system configurations have evolved, which can be classified by the movement space of the vehicles. In the most common configuration, the vehicles are restricted to a single storage aisle and tier. By contrast, in other configurations, the vehicles are able to change tier by using lifts and move between storage aisles by using cross aisles, which are positioned orthogonally to the storage racks.

Figure 1 provides an overview of the four configurations that result from different movement spaces of the

vehicles. The x-axis corresponds to the storage aisles, the y-axis to the lifts, and the z-axis to the cross aisles.

Degree of freedom	Characteristics			
Change of an aisle	not possible		possible	
Change of a tier	not possible		possible	
Configuration	aisle-captive tier-captive	aisle-to-aisle tier-captive	aisle-captive tier-to-tier	aisle-to-aisle tier-to-tier
Movement axes	x	x / y	x / z	x / y / z

Figure 1: Shuttle System Configurations

Shuttle systems with aisle- and tier-captive vehicles provide the highest throughput, as the vertical and horizontal movements are completely decoupled from each other. Shuttles hand over the storage units to buffer locations and do not have to wait for the lifts.

But as the number of degrees of freedom increases, the flexibility of the system improves. One important characteristic of shuttle systems with aisle-to-aisle and tier-to-tier vehicles is that every vehicle can reach every single position within the storage system. Therefore it is possible to run the whole system with a single shuttle. If needed, the number of shuttles can be gradually increased in order to achieve a higher throughput. As every storage unit can be delivered to every lift and therefore to every input/output location (I/O location), no merges are required within the pre-storage area. Furthermore, the storage units can be delivered in the desired sequence at every I/O location.

However, such a configuration requires a more complex control strategy in order to run the system robustly and efficiently. The main issues that have to be addressed by the control are dispatching and routing: Where and when should a vehicle travel? And which route should be taken to reach a designated position?

In our research we investigate these questions, focusing on shuttle systems with aisle-to-aisle and tier-to-tier configurations (see figure 2), and evaluating the developed strategies based on simulations.

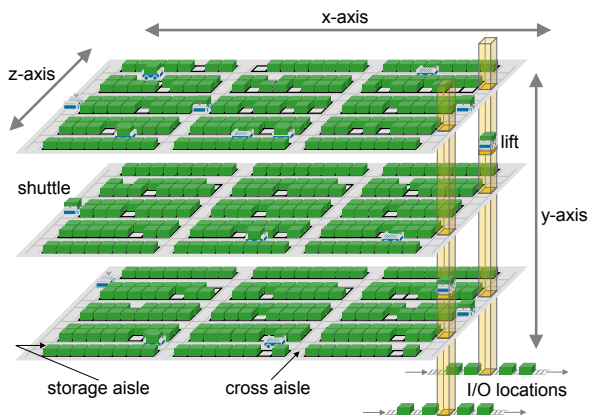


Figure 2: Example of a System

The routing must not only choose the routes themselves, but must also take deadlocks into account, since several shuttles are moving within the same rail system. The notion of deadlock describes a situation where one or more concurrent processes in a system are blocked forever because the requests for resources by the processes can never be satisfied (Kim et al. 1997). In the context of routing vehicles, the processes correspond to the execution of each route and the resources correspond to the layout segments along these routes.

Three generic approaches can be distinguished in deadlock handling: static deadlock prevention; detection and recovery; and dynamic deadlock avoidance, which generally allows the highest resource utilization (Liu and Hung 2001). One possibility for avoiding deadlocks is the time window routing method, which was introduced by Kim and Tanchoco (Kim and Tanchoco 1991). The main idea of this approach consists in modelling the layout as a graph. For each node, the algorithm maintains a list of time windows reserved by routed vehicles and a list of free time windows in which vehicles can be routed.

Since the algorithm is guaranteed to find the fastest deadlock-free path for the vehicles from their start node to their destination node at the specified start time, it is a promising approach. Adapting this algorithm for routing shuttles seems worth investigating and is the object of this paper.

## LITERATURE REVIEW

In this section, we give a review of the research into control strategies for tier-to-tier and aisle-to-aisle configurations, followed by a presentation of some applications that implement the time window routing method in different logistical contexts.

Most of the recent research concerning AVS/RS considers tier- and aisle-captive configurations and investigates the performance of the considered system in terms of parameters such as the number of storage bays and tiers or different velocity profiles for the vehicles and lifts. Only a few papers consider control strategies for tier-to-tier and aisle-to-aisle configurations.

Ekren et al. (Ekren et al. 2010) investigated the effect of several design factors on the performance of the storage system. The authors varied the dwell point and the I/O location and used basic dispatching rules to assign storage or retrieval transactions to vehicles.

Roy et al. (Roy et al. 2014) developed simple protocols to cope with deadlocks. If there is a deadlock within a storage aisle, the rearmost vehicle travels to the last available bay location where it waits until the other vehicle completes its task in that aisle. These blocking effects were quantified by an analytical model based on queuing network theory. Their strategy is not robust, as several vehicles can enter into a deadlock, and the applicability is limited to the specific layout that they considered.

Penners (Penners 2015) examined a simplified, isolated tier of an aisle-to-aisle system. He adapted two deadlock-avoiding routing algorithms and compared the performance by conducting a simulation study. He came to the conclusion that the time window routing method, modified by ter Mors et al. (ter Mors et al. 2007), achieves a considerably higher throughput than modified Banker's routing, which was described by Kalinovic et al. (Kalinovic et al. 2011). He modelled the tier as a graph, where the nodes represent lifts, lift buffers, storage aisles and crossing aisles. The use of the nodes is exclusive, which means that only one shuttle can occupy any given storage aisle at the same time, which does not take into account the real size of the aisles.

The concept of time window routing, which showed promising results, was first introduced by Kim and Tanchoco (Kim and Tanchoco 1991) for the conflict-free routing of automated guided vehicles in a bidirectional network whose nodes represent important locations such as load transfer stations, parking lots, and battery charging stations. These nodes are interconnected by lanes, which are either unidirectional or bidirectional. The size of the nodes corresponds to a check zone size, which protects vehicles from collisions. The use of the nodes is exclusive, but several vehicles can move along the same edge, as long as head-on and catching-up conflicts are prevented. Maza and Castagna (Maza and Castagna 2005) considered automated guided vehicles and implemented time window routing as proposed by Kim and Tanchoco. They developed a procedure to avoid deadlocks in the presence of interruptions while maintaining the planned routes. They showed that the absence of deadlocks is guaranteed if the node's crossing order of the vehicles based on the conflict-free scheduled date is fulfilled, even if the arrival dates are not.

Busacker (Busacker 2004) developed a time-window-based routing algorithm for optimizing aircraft taxi traffic at airports. The airport is modelled as a graph whose edges correspond to taxiways, parking positions, or any other locations that aircraft might occupy. The edges are weighted by the travel time of the airplanes and connected by nodes that do not have any physical size. As the use of the edges is exclusive, long edges are

divided into several shorter edges in order to obtain a higher utilization of resources. The size of the airplanes is not modelled.

Stenzel (Stenzel 2008) used the time window routing approach to route automated guided vehicles within container terminals. The moving area is modelled by a grid graph. Routes are computed in two steps. Firstly, a time window route is calculated, followed by a readjustment that takes into account the real size of the vehicles.

Ter Mors (ter Mors 2010) presented a generic model for routing agents through an infrastructure graph. To calculate the route, a resource graph is generated whose nodes correspond to the nodes and edges of the infrastructure graph. The edges of the resource graph can be interpreted as a successor relation. His version achieves better worst-case performance than the original algorithm by Kim and Tanchoco and calculates a solution in real time.

In summary, routing and deadlock handling for aisle-to-aisle configurations have so far only been considered in a few papers, and only in idealized terms that do not allow the developed strategies to be applied efficiently to real systems. It has been shown that the time window routing method is a promising approach that has so far been successfully applied to different contexts, and diverse answers have been given to the question of how the infrastructure of the considered system can be modelled as a graph. We will apply the time window routing method to shuttle systems. We therefore adapt the generic version by ter Mors. His approach is described in the following section.

### TIME WINDOW ROUTING METHOD

The time window routing method is used to obtain deadlock-free routes for vehicles moving through infrastructure modelled by a graph. For each node, the algorithm maintains a list of free time windows through which vehicles can be routed. Each free time window is defined by the end of the preceding reserved time window and the beginning of the subsequent reserved time window, apart from the final free time window at each node, which is endless (see figure 3).

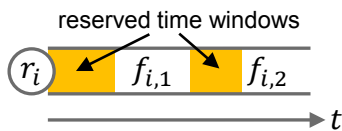


Figure 3: Free Time Windows  $f_i$  on the Node  $r_i$

The algorithm inputs consist of the start node  $r_1$ , the start time, and the destination node  $r_n$ . The output is a plan  $\pi$ .

$$\pi = (\{r_1, [t_1, t'_1]\}, \dots, \{r_n, [t_n, t'_n]\}) \quad (1)$$

The plan  $\pi$  (1) contains all the nodes along the fastest path  $r_1, \dots, r_n$  and the corresponding time windows,

which are defined by the entry times  $t_i$  and the exit times  $t'_i$  at the nodes  $r_i$ .

The algorithm consists of two consecutive steps that are executed iteratively. These steps are:

1. Investigate the reachability of all free time windows on all neighbouring nodes.
2. Select the most promising time window for the next iteration.

Starting with the initial time window on the start node, each iteration of the algorithm investigates the reachability of all free time windows on all neighbouring nodes (see figure 4). This procedure is called time window expansion.

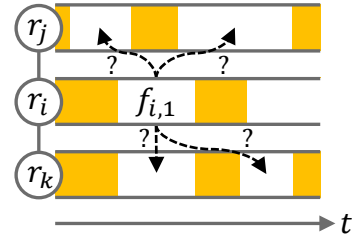


Figure 4: Time Window Expansion

In order for a free time window to be reachable, some conditions must be met. A free time window on a neighbouring node is reachable from the current free time window if:

- the free time window is larger than the minimal duration required for a vehicle to enter, traverse, then exit the resource again,
- both free time windows overlap,
- the current free time window can be completely exited before it ends,
- the remaining duration after entering the free time window is large enough to traverse and exit the resource.

If a free time window is reachable, it is added to a list called open list, which contains the time windows for further expansions.

In the next iteration, the most promising time window is selected from the open list and removed. The most promising time window is the one that allows the final destination to be reached in the theoretical minimal time. This time  $y(f)$  is the sum of earliest possible exit time  $c(f)$  from the current node and the estimated time  $h(f)$  required to complete the routing to the destination:

$$y(f) = c(f) + h(f) \quad (2)$$

The algorithm terminates as soon as a free time window belonging to the destination node is selected for the next iteration. Once this happens, the plan  $\pi$  is constructed using back pointers and the corresponding time windows are reserved.

The example in figure 5 shows a graph and the fastest path through the free time windows from the start node  $r_1$  to the destination node  $r_5$ . The overlapping between free time windows is illustrated. Note that the fastest path might visit a node twice.

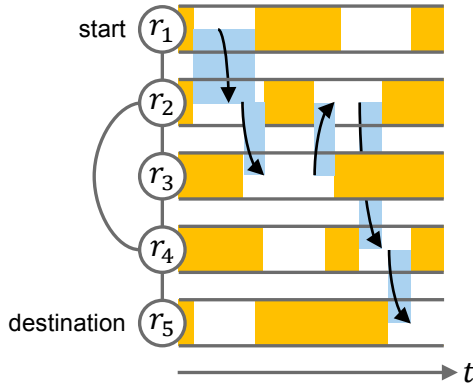


Figure 5: Fastest Path from the Start Node  $r_1$  to the Destination Node  $r_5$

If a free time window needs to be selected for the next iteration from the open list but the open list is empty, there is no route available. This can happen if the time window on the start node is not endless or if vehicles are allowed to dwell on nodes indefinitely.

The search for the fastest path through the free time windows is an application of the A\* algorithm, which is used to find a shortest path in a graph (Hart et al. 1968). Each free time window corresponds to a node in a so-called free time window graph. The arcs between these nodes represent the reachability between pairs of free time windows. The algorithm runs in polynomial time in the size of this time window graph. Hence, assuming a feasible number of nodes in the resource graph and a feasible number of vehicles, the algorithm returns a solution in real time.

As a closing remark, we should note that the sequence in which vehicles are routed matters. The algorithm finds an optimal solution for a single vehicle under the given reserved time windows but does not find the global optimum (e.g. the makespan). In the shuttle system, the most commonly encountered process is that of a single shuttle that occasionally arrives on a storage tier needing to be routed. Apart from initialization, it is unlikely that a larger number of shuttles will need to be routed at the same time. Hence the presented approach can be used for our purposes.

In the remainder of this paper, we model the tier of the shuttle system as a graph in order to apply the time window routing method, which allows high resource utilization to be achieved. We must therefore answer the question of what the nodes and edges should represent, and how their sizes should be determined.

Below, we modify the time window routing method proposed by ter Mors (ter Mors 2010) in order to use it to route the shuttles on a storage tier.

Finally, we present a control strategy that allows the calculated routes to be executed even if there are delays and uncertainties.

## MODELLING THE SYSTEM

The components of a shuttle system are the storage rack, the rail system, the lifts, and the shuttles. In order to apply the time window routing method, the rail system must be modelled as a graph. We apply the concept of the resource graph, which means that the edges simply represent a successor relation and do not have any physical size. The nodes are divided into the following types (see figure 6):

- Storage aisle nodes  
Storage aisle nodes are placed within a storage aisle. In order to access a storage location, shuttles must occupy these nodes.
- Cross aisle nodes  
Cross aisles are positioned orthogonally to the storage racks. The nodes are placed between the crossing nodes and are used to travel between storage aisles.
- Crossing nodes  
Crossing nodes interconnect at least three aisle nodes and allow a shuttle to perform a 90 degree turn in order to move from a cross aisle into a storage aisle or vice versa.
- Buffer nodes  
Buffer nodes are placed at the edges of storage aisles or cross aisles and are used for buffering idle shuttles. This prevents idle shuttles from blocking other shuttles that are performing a storage or retrieval transaction.
- Lift buffer nodes  
Lift buffer nodes are placed on both sides in front of the lifts and are used as input and output buffers on every storage tier.
- Lift nodes  
Lift nodes represent the lifts. They can be entered only if the lift is empty and is currently located on that tier.

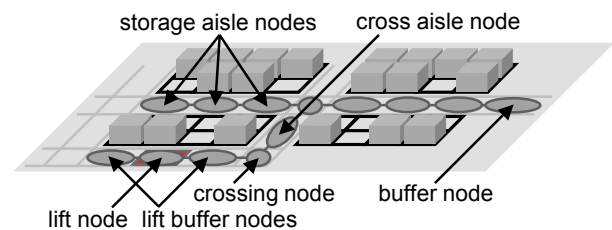


Figure 6: Different Types of Nodes on a Storage Tier

Except for the lift buffer nodes, all nodes can be traversed in both directions. We therefore use a mixed graph in which only the edges incident to the lift buffer nodes are directed, since lifts are accessed from one side and exited from the other side, as this allows a higher throughput to be achieved.

Every node has an attribute that represents the axis along which the vehicle is travelling at that node. For instance, aisle nodes allow only movement along the x-axis and cross aisle nodes allow only movement along the z-axis. Crossing nodes allow both orientations. Whenever a time window is expanded to a time window on a crossing node, the orientation of the vehicle entering the crossing node is stored. If the subsequent node has a different orientation, a 90-degree turn on the crossing node is necessary. In order to model the rail system, we enforce the following conditions:

1. The use of each node is exclusive.
2. The size of any given node must not be smaller than the size of the shuttle.

The first condition ensures that no sub-routing is required on single nodes, as only one shuttle at a time is allowed to move on the node. The second condition avoids having to take into account more than two nodes for the reachability check, which would considerably increase the complexity of the algorithm. To achieve high resource utilization, the nodes should be as small as possible.

The size of the buffer nodes, lift nodes and cross nodes corresponds to the layout data, as we assume only one shuttle can occupy these nodes. The length  $L_{Aisle}$  of both the storage aisles and cross aisles between two crossing intersections is also given, as well as the length of the shuttles  $L_{Shuttle}$ , which could be replaced by the minimum node length. The number  $N_{nodes}$  of nodes within a storage or cross aisle and their lengths  $L_{nodes}$  are calculated as follows:

$$N_{nodes} = \text{floor}\left(\frac{L_{Aisle}}{L_{Shuttle}}\right)$$

$$L_{nodes} = L_{Shuttle} + \frac{L_{Aisle} - N_{nodes} * L_{Shuttle}}{N_{nodes}}$$

If a shuttle needs to be buffered or needs to use a lift, it will be routed to the corresponding node. By contrast, the aisle nodes do not represent destinations in the system themselves.

In order to fulfil a storage or retrieval request, a shuttle must be routed to a storage location. Since the size of the shuttles exceeds the size of the storage units and therefore the size of the storage locations, the size of the storage locations is smaller than the length of the aisle nodes.

In the example shown in figure 7, the storage aisle consists of four nodes. In order to access a storage location, for some storage locations it is sufficient to reserve a single node (e.g. storage location no. 2), whereas for some storage locations two nodes must be reserved (e.g. storage location no. 9).

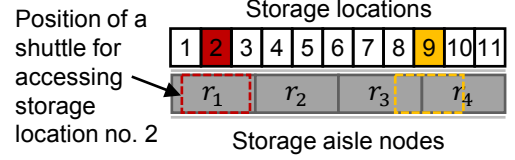


Figure 7: Storage Aisle Nodes

The nodes that have to be reserved in order to perform a storage or retrieval transaction can be easily identified by considering the shuttle's exact position when accessing the location.

## MODIFICATIONS TO THE ALGORITHM

In order to use the described time window routing method, some modifications are necessary, which we will describe below.

1. We select the most promising time window for the next iteration in a slightly different way, as we cannot assign an earliest possible exit time to a time window.
2. The algorithm does not necessarily stop as soon as the destination node is reached for the first time, since the time window must guarantee a minimal remaining size after entrance.
3. Instead of a single destination node, we use a set of destination nodes, as it might be necessary to reserve several nodes in order to access a storage location.

Recall that the most promising time window is identified by the value

$$y(f) = c(f) + h(f) \quad (2)$$

We cannot associate an earliest possible exit time with each time window, because this time also depends on the orientation of the neighbouring node. If a 90-degree turn is necessary, the exit time is postponed. Hence, for the expression  $c(f)$ , we use the arrival time at the node, which describes the moment when the shuttle is located at the centre of the node. Furthermore, to estimate the remaining travel time  $h(f)$ , we use the fastest possible path from the current node (and orientation) to the destination node without any reservations within the system.

The described version of time window routing terminates as soon as a time window belonging to the destination node is selected. For our purposes, in some cases the time window must be large enough to guarantee a minimal remaining length. Shuttles travel across each storage tier in order to perform storage or retrieval transactions. Each of these transactions requires a certain amount of time. Therefore, if a storage or retrieval transaction needs to be fulfilled at a destination node, the remaining size of the time window must allow

the performance of the transaction and full exit from the node.

Instead of a single destination node, we use a set of destination nodes called the destination set. This set usually contains a single node, but when routing towards the storage locations, it contains all the nodes that must be reserved in order to perform the storage or retrieval transaction. Hence, the set contains either one or two nodes. As soon as a time window belonging to the destination set is selected for the next iteration, the algorithm checks whether there is another node in the set. If there is, there must be a free time window on that node that is reachable from the current time window. The overlapping of these two time windows must not only allow the movement of the vehicle from one node to the other, but also must be long enough to allow the storage or retrieval transaction to be performed. If no time window is reachable on the other node, or the remaining size of the time window does not allow the storage or retrieval transaction to be performed, the search through the free time windows is continued until a later free time window is found that guarantees these conditions.

## IMPLICATIONS FOR THE CONTROL

The result of the routing process is a list with the nodes that have to be visited in order to reach all designated destinations on a tier and the corresponding time windows during which shuttle will move to and occupy these nodes. As the route is calculated using idealistic times and neither accelerations and decelerations are fully taken into account, it is not sufficient to move the shuttles according to their reserved time windows. Furthermore shuttles are routed to the lifts. But when the routing is calculated, it is not obvious when the shuttle will enter the lift and exit the previous node, as the control of the lifts is decoupled from the routing. Consequently, delays will be passed from one node to another on each storage tier.

Therefore it is not possible to navigate the shuttles by the time windows alone; only according to the sequence of reserved time windows on the nodes. As Maza and Castagna (Maza and Castagna 2005) showed, the absence of deadlocks is guaranteed as long as the nodes' crossing order is preserved.

Hence we introduce the concept of claiming nodes. A shuttle is allowed to enter a node only if it has previously claimed that node. A node can be claimed by a shuttle if and only if the earliest reserved time window on that node was reserved by the shuttle. Therefore a node cannot be claimed by several shuttles simultaneously.

In order to clarify this process, we consider the following simplified example, shown in figure 8, with three shuttles that must be routed.

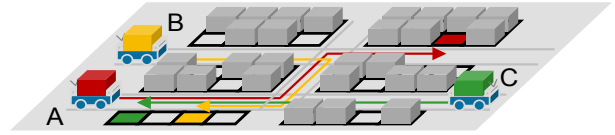


Figure 8: Routing on a Storage Tier

Firstly, shuttle A routes and reserves its time windows, followed by shuttle B and shuttle C. Note that shuttle B has to wait a certain period of time on node  $r_{14}$  before it can enter node  $r_{15}$ , and shuttle C also has to wait on node  $r_6$ .

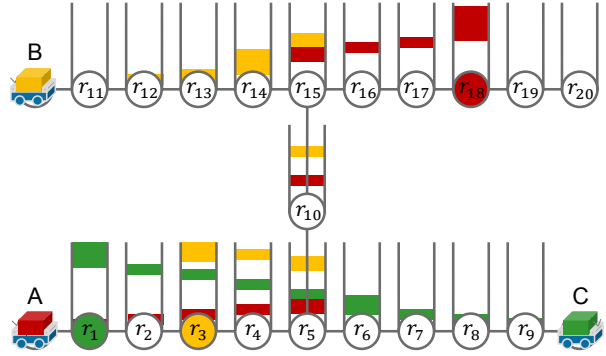





Figure 9: Reserved Time Windows by the Shuttles

For each node, we know which shuttle will occupy the node during each time period. We therefore also know the sequence of shuttles that will cross this node. If we consider the reservation list of crossing node 5, we get the following information.




Table 1: Reserved Time Windows on Node 5

Shuttle	Entry Time	Exit Time
A 	00:00:08	00:00:12
C 	00:00:12	00:00:15
B 	00:00:17	00:00:21

At this node, the sequence of shuttles is A, C, B. This sequence has to be established so that deadlocks can be avoided even if the shuttles are late for some reason.

Whenever a shuttle starts moving, we identify the nodes that could be traversed by the shuttle. These nodes are then claimed by the shuttle. In the example, the following nodes are claimed.

Table 2: Nodes Claimed by the Shuttles

Shuttle	Claimed Nodes
A 	$r_1, r_2, r_3, r_4, r_5, r_{10}, r_{15}, r_{16}, r_{17}, r_{18}$
B 	$r_{11}, r_{12}, r_{13}, r_{14}$
C 	$r_9, r_8, r_7, r_6$

Shuttle A was able to claim its whole route, whereas shuttle B and C could claim only part of their routes. If a shuttle was able to claim at least one other node, it starts moving and stops as soon as it finishes a claimed segment. It will then start claiming the next segment of nodes again.

If a shuttle cannot claim a single node, it will register as waiting shuttle at this node. Whenever a shuttle exits a node completely, not only is the reserved time window deleted, but the algorithm also checks whether any other shuttle is registered as waiting for that node. If another shuttle is registered on the node, it is triggered and will claim its next segment.

The concept of claiming nodes is necessary. Examining how far a shuttle is allowed to travel is insufficient, since another shuttle can reserve the earliest time window at any moment. The shuttle that had previously reserved the earliest time window would then no longer be allowed to enter that node. Therefore, whenever the earliest time window is reserved on a node, the algorithm checks whether that node has already been claimed by another shuttle. If it has, the node is released by that shuttle, as well as all subsequent nodes along the route that have been already claimed.

## SUMMARY

In this paper we described shuttle systems as a technology for storing small unit loads. We focused on systems with a tier-to-tier and aisle-to-aisle configuration, which provide high flexibility. In these systems, every vehicle can reach every storage location. From the perspective of control, routing becomes an important issue due the possibility of deadlocks among the shuttles, which must be dealt with.

As a concept for routing and handling deadlocks, we referred to the time window routing method that has already been successfully applied in different logistical contexts. We adapted the time window routing method and modelled the tier of the shuttle system as a graph in order to apply the method. Finally, we described a concept that enables the vehicles to execute the calculated routes.

The time window routing method was implemented in a generic simulation model kit for shuttle systems, as well as the concept of claiming nodes. In future work, these concepts will be evaluated by simulation experiments.

The concept of claiming nodes could be expanded. It might be possible to allow nodes to be claimed by shuttles that did not reserve the earliest time window on these nodes if this improves the overall efficiency and the absence of deadlocks can be guaranteed. Furthermore, fully integrating the lifts into the time window routing scheme might be interesting.

## REFERENCES

Busacker, T. 2005. *Steigerung der Flughafen-Kapazität durch Modellierung und Optimierung von Flughafen-Boden-Rollverkehr – Ein Beitrag zu einem künftigen*

- Rollführungssystem*. Dissertation. Technische Universität Berlin.
- Ekren, B. Y., Heragu, S. S., Krishnamurthy A. and Malmberg C. J., 2010. "Simulation based experimental design to identify factors affecting performance of AVS/RS." *Computers & Industrial Engineering* 58, No.1, 175-185.
- Hart, P. E., Nilsson, N. J. and Raphael, B. 1968. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths" *IEEE Transactions of Systems Science and Cybernetics* 4, No.2, 100-107.
- Kalinovic, L. , Petrovic, T. , Bogdan, S. and Bobanac V., 2011. "Modified banker's algorithm for scheduling in multi-agv systems." *Automation Science and Engineering* (Trieste, Italy, Aug. 24-27), 351-356.
- Kim C. W. and Tanchoco J. M. A., 1991. "Conflict-free shortest-time bi-directional AGV routing." *International Journal of Production Research* 29, No.12, 2377-2391.
- Kim C. W., Tanchoco J. M. A. and Koo P., 1997 "Deadlock Prevention in Manufacturing Systems with AGV Systems: Banker's Algorithm Approach." *Journal of Manufacturing Science and Engineering* 119, No.4, 849-854.
- Liu F. and Hung P., 2001. "Real-time deadlock-free control strategy for single multi-load automated guided vehicle on a job shop manufacturing system." *International Journal of Production Research* 39, No.7, 1323-1342.
- Malmberg C. J., 2002. "Conceptualizing tools for autonomous vehicle storage and retrieval systems." *International Journal of Production Research* 40, No.8, 1807-1822.
- Maza, S. and Castagna, P., 2005. "A performance-based structural policy for conflict-free routing of bi-directional automated guided vehicles." *Computers in Industry* 56, No.7, 719-733.
- Penners L. T. M. E., 2015. *Investigating the effect of layout and routing strategy on the performance of the Adaptive system*. Master's Thesis. Eindhoven University of Technology.
- Roy, R., Krishnamurthy, A. and Heragu, S. S. 2014. "Blocking Effects in Warehouse Systems With Autonomous Vehicles." *IEEE Transactions on Automation Science and Engineering* 11, No. 2, 439-451.
- Stenzel, B. 2008. *Online Disjoint Vehicle Routing with Application to AGV Routing*. Dissertation. Technische Universität Berlin.
- ter Mors, A. W., Zutt, J. and Witteveen C., 2007. "Context-Aware Logistic Routing and Scheduling." In *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling* (Providence, USA, Sep. 22-26), 328-335.
- ter Mors, A. W. 2010. *The world according to MARP*. Dissertation. Technische Universiteit Delft
- VDI-Richtlinie 2692 Blatt 1, 2015. Automated vehicle storage and retrieval systems for small unit loads. Berlin: Beuth.

**THOMAS LIENERT** has been working as a research assistant at the Institute for Materials Handling, Material Flow and Logistics, Technical University of Munich, since 2014. His research deals with the development of control strategies for autonomous vehicle-based storage and retrieval systems. His email address is: [lienert@fml.mw.tum.de](mailto:lienert@fml.mw.tum.de).

**JOHANNES FOTTNER** is professor and head of the Institute for Materials Handling, Material flow, Logistics at the Technical University of Munich.