STAIRS 2012
K. Kersting and M. Toussaint (Eds.)

doi:10.3233/978-1-61499-096-3-270

Towards Decentralised AGV Control With Negotiations

Christoph SCHWARZ [a,1] and Jürgen SAUER [b]
[a] *OFFIS - Institute For Information Technology, Oldenburg, Germany*
[b] *Carl von Ossietzky University, Oldenburg, Germany*

Abstract. Flexible solutions in the field of intralogistics are needed more and more because of a higher dynamic in the environment. A possible solution to achieve higher flexible and are robuster system is the use of an automated guided vehicle (agv) system. In this paper current research in the field of decentralized agv system control i.e. in conflict free routing and in decentralized order allocation is described. Furthermore approaches to improve self-control by introducing negotiations are presented as well as the challenges that arise when such approaches should be realized.

Keywords. self-control, multi agent system, negotiations, intralogistics, automated guided vehicles

Introduction

Efficient ways to control intralogistics become more and more important. As a reason for this observation Follert and Roidl stated ”Intralogistics is a cutting-edge term in Europe that comprises all technical systems, services and related business involved in the in-house materials handling of industrial enterprises, wholesalers, retailers and government institutions. The processes of the intralogistics domain are vital for managing the flows of goods along the entire supply chain as they provide the reliable and predictable flow of physical goods in the joints of a supply network.” (from [1]). Tompkins et. al. showed that intralogistics fill a key position between engineering and economy (see [2]).

Simultaneously with the increasing importance of intralogistics the complexity of these processes have increased significantly in the last decades (see [3]). Traditionally logistic and intralogistic systems rely on a strict hierarchy. Such a hierarchy control has however some serious drawbacks. Versteegt stated that ”Hierarchical systems are rigid and static; modifications are hard to incorporate and costly, and hierarchical systems cannot cope effectively with disturbances.” (from [4]).

These two factors have lead to increased research into decentralized control mechanisms since a decentralized control promises more flexibility. This demand is mentioned for example by Schreiber and Fay: ”Today’s manufacturing systems show an increasing level of complexity and require more flexibility with respect to an increasing volatile process environment. For this, advanced concepts for manufacturing control are necessary,

[1] Corresponding Author: Christoph Schwarz; E-mail: Christoph.Schwarz@offis.de
Christoph Schwarz is currently a doctoral candidate supervised by Prof. Jürgen Sauer.

as for example multi-agent systems (MAS)." (from [5]). Also Scholz-Reiter et. al. points out the reason for using autonomous control: "The use of autonomous control aims at a higher robustness of systems and simplified processes achieved by distributed handling of dynamics and complexity due to greater flexibility and autonomy of decision making" (from [6]).

In the field of intralogistics often automated guided vehicles are used to achieve flexibility. Singh et. al. stated: "Automated guided vehicles (AGVs) are being increasingly used for material transfer in production lines of modern manufacturing plants. The purpose is to enhance efficiency in material transfer and increase production" (from [7]).

In this paper we want to focus on the two main aspects of decentralized control of an intralogistic agv system: Conflict free routing and decentralised order disposition. Both aspects should be combined with negotiations between the agents to improve the global system behaviour.

Paper Organisation

The remainder of the paper is structured as followed: In the next section the problem descriptions of conflict free routing and decentralized order allocation as well as related work from these fields are presented. In the sections thereafter we will present the approaches we plan to investigate (section 2), the main challenges that arise with these approaches (section 3) and the planned evaluation (section 4). Finally in the section conclusion a brief overview is given.

1. Problem Description and Related Work

Decentralised control of agv systems is in the scope of research for several years. In this section the problem descriptions and accomplishments in the field of decentralized order disposition and in the field of conflict free routing are described.

1.1. Conflict free routing

In autonomous controlled agv systems the individual agvs have to plan their route through the area (for example the warehouse) by their own. But since there are other agvs in the area the risk of conflicts arise. For example see figure 1. In this situation a conflict between agv 1 and agv 2 would arise at the section of the layout which is marked with a red circle if both agvs would plan without considering the plan of the other vehicle. If such a conflict occurs a time-consuming conflict resolution have to be done (for example one agv has to drive backwards to let the other one pass). To avoid such conflicts and the resulting resolutions a form of conflict free routing is necessary.

The concept of conflict free shortest path routing for agvs was introduced in 1985 by Broadbent et. al. Their method presented in [9] uses the Dijkstra algorithm (see [10]) to calculate a matrix which represents the assignments of nodes by the vehicles. In this matrix initially shortest paths for all vehicles are calculated. In a second step possible conflicts are solved by slowing some vehicles down or plan alternative routes for some vehicles. It is therefore a two-step process. First, an initial set of routes is calculated and in the second step this set is checked for conflicts.

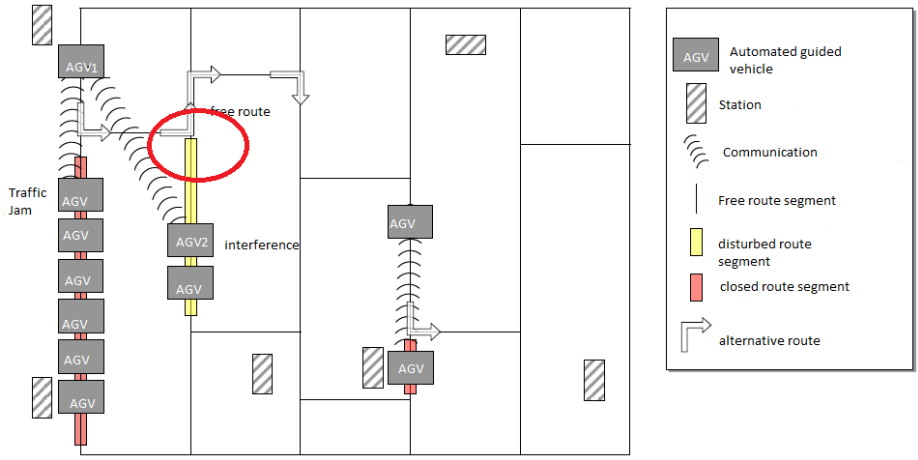


Figure 1. Conflicts in routing

1991 Kim and Tanchoco presented in [11] the concept of the time window graph. In the time window graph the nodes represent the free time windows and the edges represent the accessibility between the nodes. This graph is constructed out of the static way graph and the dynamic information about already planned routes. This time window graph is dynamic in the sense that it will be recalculated after any change in the planned routes. After the calculation of this time window graph the Dijkstra algorithm is used to find a shortest path in this graph. In order to enable the vehicles to wait at some node as part of their plan the graph contains loops. The entire process has an asymptotic running time of $o(v^4n^2)$ where v corresponds to the number of vehicles and n to the number of nodes in the given way graph. The method determines, in contrast to the method of Broadbent et.al., conflict free routes in one step.

Möhring et.al. present in [12] an approach to conflict free agv routing which allows an online computation even with dynamic orders and consideration of the physical dimensions of the vehicles. The proposed algorithm prevents collisions, deadlocks and lifelocks already in the planning and takes into account the physical dimensions of the vehicles. Therefore for each new request (ie each new incoming transfer order) as a first step polygons $P(a)$ for each arc a which represents the blocked parts of the arc (blocked by vehicles which plan to use these arcs) are calculated. After this preprocessing the physical dimensions of the vehicles can be ignored during the calculation of a shortest path. The original problem is thus reduced to a shortest path problem with time windows (SPPTW - see [13]). The general form of the SPPTW is NP-complete (see [12]) but Möhring et.al. show that the SPPTW is solvable in polynomial time when only transit times (including waiting times) are used as cost function.

2007 ter Mors et. al. presented a free path routing approach named context-aware routing which solves the conflict free routing problem with an asymptotic running time of $O(nv\ log(nv) + nv^2)$ where v corresponds to the number of vehicles and n to the number of nodes in the given way graph (see [14]). In their case the agents reserve the way segments they plan to use and thus alter the conditions for agents which are planing after them (because more reservations are in the system). When agents plan a new route they are not allowed to violate reservations (i.e. to use way segments on time intervals for

which they are reserved by other vehicles). To solve the planing problem Mors et.al. build the following model: A set $\mathscr{A}$ of agents, a set $\mathscr{R}$ of resources (for example way segments or elevators or battery loading stations) where each resource $r_i \in \mathscr{R}$ has a capacity $C(r_i)$ (which indicates how many agents can use the resource simultaneously), a successor relation $\mathscr{S} \subseteq \mathscr{R} \times \mathscr{R}$ (indicating weather you can get one from one resource to another) and the values $D(r_i)$ indicating how long it takes to traverse the resource r_i. A plan of an agent then consists of a a sequence of resources and the time intervals in which they are visited: $P = ((r_1, \tau_i), \ldots, (r_n, \tau_n))$. The following conditions hold for a plan to be feasible: A plan is not allowed to have gaps or double assignment in the sequence of time intervals, the time interval planned to traverse a resource r_i must be as least as great as $D(r_i)$ and for two succeeding resources r_i, r_j $r_i \times r_i$ has to be element of $\mathscr{S}$ (meaning r_j is reachable from r_i). With these definitions the dynamic free time windows can be defined as follows: a free time window on a resource r_i is a time interval which does not overlap with a reservation on this resource and which is as least as long as the minimal traversal time $D(r_i)$. When an agent wants to plan a route the set of free time windows is calculated for each resource. These free time windows are the nodes of the time window graph. Afterwards the edges are created. An edge between two free time windows is added if: the corresponding resources of the free time windows are in $\mathscr{S}$ and when there is enough time to traverse the resources in the free time windows. Finally a shortest path is calculated in the free time window graph. In this graph resources can be visited multiple times (up to one time per free time window the resource has).

Besides the development of new algorithms for conflict-free routing, there are also other approaches. For example in [7] Singh et.al. present a method that partitions the entire layout in so called exclusive zones and shared zones. This partitioning is based on observations of the need for agvs at certain points of the layout (storage areas, processing machnies, etc.). The method has been especially designed for layouts where there are no alternative routes. After the partitioning each of the exclusive zones is assigned to exactly one agv (an agv can be assigned to more than one zone). The assignment (and the partitioning) must be done in such a way, that the transport capacity of the agv is sufficient for the demand of the zone. In addition to this necessary condition it is tried to keep the shared areas of the layout (the areas which more than one agv uses) as small as possible and thus minimize the conflict risk. To avoid conflicts and deadlocks eventually all vehicles have to reserve the shared areas. If another vehicle uses a shared zone other vehicles wait in their exclusive zones for that other vehicle to leave the shared zone before they enter that zone.

Another method was presented by Smolic-Rocak et.al. in [15]. In this process, called time window based dynamic routing, a set of short paths is calculated offline (ie in advance) for every node pair (ie combinations of all possible start and end points of transfer orders). This is done by a shortest path algorithm like the Dijkstra algorithm (see [10]). When in operation a new mission, that is a new transport or driving task, is generated it is checked weather these paths are allowed with the current reservations in the system. Afterwards the shortest out of the allowed paths is chosen and reserved by the planing agent. The tests for admissibility works with time windows similar to the ones described by ter Mors et.al. (see above). For each edge a lower bound for the traversal time is given through the length of the edge an the maximum speed of the vehicles. As a first step all edge traversals are planned with these lower bounds. Afterwards it is checked if the resulting route has any conflicts with already planned routes. If there are such conflicts it

is tried to circumvent the first conflict by waiting on the edges before this conflict. This process is iterated until all conflicts are resolved (which lead to a feasible plan) or until the total time needed to execute the current plan exceeds the time that is available for fulfilling the order (which means the path is infeasible).

1.2. Decentralized and dynamic order allocation

The decentralized allocation of orders is a important task in nearly any decentralized system (see for example [16]). The purpose of task allocation is to assign the tasks that occur in the system to the different entities that could handle them. In decentralized order allocation there is no global entity which assigns all of the orders thus the autonomous units have to solve this problem by their own. A decentralized order allocation becomes dynamic if the assignments can change over time (meaning a once made (temporal) assignment can be revoked).

The decentralized order allocation is often solved with heuristic approaches and with approaches from the field of game theory (see for example [17]). The techniques usually presumes that the set of possible task, the set of agents and the results of each possible assignment is known in advanced. Therefore these techniques seem to be not usable for dynamic order allocation.

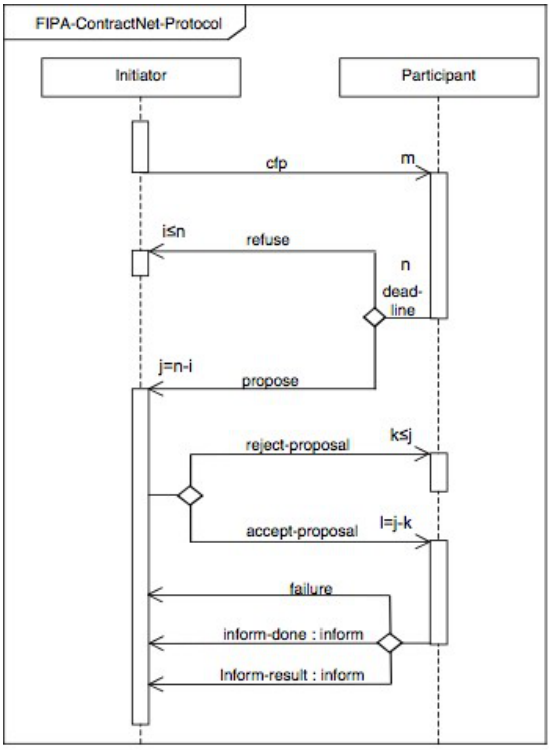


Figure 2. FIPA-ContractNet-Protocol from [8]

Another set of solutions are auction like mechanisms. Among the wildest known is the Contract Net (CNET) proposed by Smith (see [18]). In the CNET protocol some

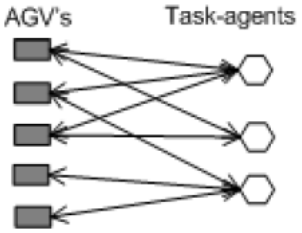


Figure 3. Many-to-many communication relation (from [20]

agents have the role of managers (or initiators), some the role of bidders (or participants). Figure 2 depicts the processes in the FIPA[2] -CNET protocol. In the agv system case managers could be load stations, battery stations, production facilities, etc. and the agvs would usually have the role of the bidders. If a new order is created (e.g. a transport unit should be moved from a to b) the corresponding manager informs the bidders of this order. Than the bidders send a proposal. This proposal can be the time they would need to perform this order, or the amount of energy they would need or how many kilometres they would have to drive (or a (weighted) combination of these or other factors). After a defined weighting time the manager assigns the order to the agent (agv) which has made the best proposal.

Another set of method for decentralized order allocation is nature inspired. As an example Hadeli et. al. uses stigmergy (see [19]). In this technique information about the orders and their priorities is saved in the (virtual) environment (like pheromones). Since these stigmergies are stored just in the software representation of the environment (a physical implementation would be far to costly) this environment hat to be stored and synchronized by all agents.

Boucké et.al. presented a approach for decentralized allocation of tasks which uses a special negotiation protocol in [20]. This protocol is a cyclic protocol based on task-energy and interest-energy. Every created task is associated with an task-agent and has an initial task-energy (based for example on the priority of the task). In the beginning of a cycle the task-agents inform the agvs of the tasks and their current task-energy. The agvs than answer with the amount of energy they wants to consume (the interest-energy) ”This interest depends on the task-energy (decreases with the distance to the pick-up spot), the suitability of the AGV in performing the task, the amount of consumed-energy of the previous cycle and possible other factors inuencing the interest in performing the task.” (from [20]). Agvs which currently have no load move towards the task which interests them the most (and thus the agv will be nearer to the task at the next cycle). After a fixed time the cycle is finished. The task-agent than determines how much energy is used and add a raise-energy (which preserves the task from starvation). Please note that this approach is a many (task-agents) to many (agv-agents) protocol (see figure 3). They use their approach to overcome problems that arise in decentralized agv order allocation: ” While the AGV rides towards the pick-up spot (in the third step) many things can happen: new tasks that are better suited for this AGV can show up, e.g. being much closer, more urgent or even a task being on the way to the pick-up spot; AGVs can become unavailable, because of a failure or because they suddenly have to go in maintenance;

[2]Foundation for Intelligent Physical Agents: http://www.fipa.org/

AGVs better suited for the task, e.g. closer to the pick-up spot, with more energy left in their battery, can become available." (from [20]).

2. Own Approach

Within the scope of the research project "FTS-Selbststeuerung" ("agv-self control".) the realizability of an self-controlled agv system is studied. In this project there should be no centralized entities, neither for order allocation nor routing, nor communication. Beside developing an sufficient agent framework and efficient communication protocols also rules for routing, order allocation and conflict resolution should be developed and evaluated in real scenarios (see section 4). In this paper we describe the planned approaches to routing and order allocation.

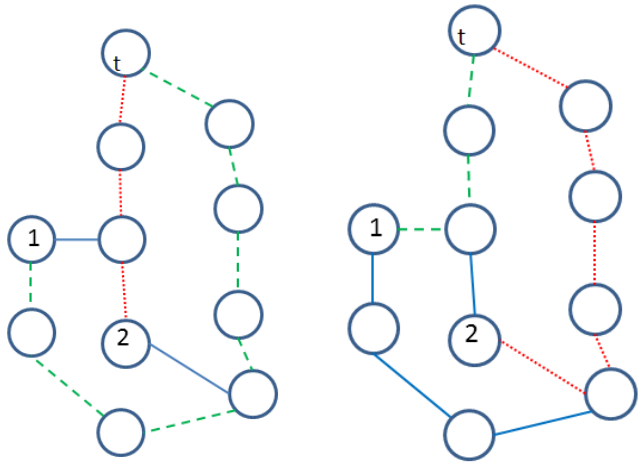
2.1. Routing

To apply techniques from subsection 1.1 to self-controlled conflict free routing it is necessary that the the agv-agents (from no one agvs since the software agent and the vehicle will be one unit) broadcast reservations they made so that other agvs can consider them when they plan routes. A message for reservation can look as follows: ($\langle$ edgeID$\rangle$,$\langle$ agvID$\rangle$,$\langle$ start time$\rangle$,$\langle$ end time$\rangle$) where start time is the time the agv enters the edge and end time the one when it is leaving the edge. Please note that due to wlan disturbances or other errors it is possible that some messages do not get through to every agv. In other words it is possible that the agvs plan with an incomplete or inaccurate reservation list.

As basis for the self-controlled conflict free routing the context-aware routing algorithm from ter Mors et.al. was chosen (see subsection 1.1). Although this approach is capable of calculating conflict free routes for every agv, ter Mors et. al. have shown in [14] that the resulting individual plan of an agv depends on the order in which the agvs make their plan. This fact is obvious since the number of reservations grows with each previous made plan. According to ter Mors et.al. the cost of the average plan increases nearly linearly with the number of reservations in the system. To overcome this fact it must be possible that reservations may be withdrawn. To allow this we want to integrate negotiations to the route planing process. To show possible benefits of negotiations consider the situation in figure 4. In subfigure (a) a situation is shown in which the agvs 1 and 2 want to get to the node labelled t. First agv 2 plans (and makes its reservations in red and dotted) than agv 1 plan (in green and dashed). The resulting plan for agv 1 has a length of 7 (assume the edges have an uniform length), the plan of agv 2 has a length of 3 leading to an average length of 5. Subfigure (b) shows the situation after agv 1 has negotiated with agv 2 and has gotten the reservations from agv 2. In the resulting situation the length of the plan of agv 1 is 3 and the length of the plan of agv 2 is 5 leading to an average length of 4. Of course this simple example does not allow to draw conclusions on a realistic scenario. We want to investigate the impact of such negotiations in realistic scenarios.

Our approach for self-controlled conflict free routing with negotiations is outlined in algorithm 1.

In a first step we calculate a shortest path ignoring reservations (p_f) and its length (c_f) with the Dijkstra algorithm (line 1). Then it is checked weather this path has conflicts



(a) Situation after conflict free routing (b) Situation after successful negotiations

Figure 4. Negotiations can lead to a better system performance

Algorithm 1 Routing

```
1:  p_f, c_f ← Dijkstra
2:  if check(p_f) then
3:      reserve(p_f)
4:      return p_f, c_f
5:  end if
6:  p_ca, c_ca ←context-aware routing (ter Mors et.al.)
7:  if c_ca ÷ c_f ≤ ε then
8:      reserve(p_ca)
9:      return p_ca, c_ca
10: end if
11: while time ≤ max response time and c_n > c_f do
12:     p_n, c_n ← negotiate
13: end while
14: reserve(p_n)
15: return p_n, c_n
```

with existing reservations (line 2) if this is not the case the path will be reserved (line 3) and used (line 4). If there are conflicts as a second step a conflict free path (p_{ca}) and its length (c_{ca}) are calculated with the contest-aware routing algorithm (line 6). Afterwards it is checked how much longer this conflict free path is compared to a shortest path (line 7). In case the length of the conflict free path is less or equal ε (we will return to ε in section 3) times the length of the shortest path it is accepted (line 9) and reserved (line 8). If the length is greater than ε times the length of a shortest path a negotiation process is started. In this process the currently planing agv tries to improves the path length through negotiations with the agvs whose reservations causes conflicts with a shortest path or with shorter paths than the conflict free one. This process is continued until either the agv

can use a shortest path or until the maximum response time is reached. The maximum response time is necessary since in a real scenario the agvs have only limited times to plan their routes. This maximum response time will depend on the scenario. Furthermore note that this process can be recursive (if vehicle x negotiates with vehicle y it can happen that vehicle y starts a negotiation with vehicle z while trying to calculate an alternative route (which it would have to use if vehicle x wins the original negotiation)). In order to guarantee the maximum response time of an initial routing this maximum response time will have to be reduced in deeper levels of the recursion.

2.2. Negotiations

In order to negotiate about reservations the agvs have to evaluate the utility they have of a certain reservation. It is planned to use parametrized scoring functions for these evaluations. In such way a agv a_n can compute the utilization of a reservation of edge e_i at time t_k as follows:

$$score_{a_n}(e_i,t_k) = p_1 \times \text{detour} + p_2 \times \text{order priority} + p_3 \times \text{order deadline} \quad (1)$$

Where p_1, p_2 and p_3 are parameters which determine the relative influence of the detour, the order priority and the deadline. Of course other influences are possible. Please note that since the agvs have to calculate the detour that would occur when they can not use the edge e_i at the time t_k they possible have to negotiate with other vehicles to find a good alternative route. To guarantee an in time response this negotiations must be done in shorter time. With the help of the parameters it should be possible to set preferences in the global system behaviour like as least driven kilometres as possible, no delays, or faster completion of high priority tasks. Besides the evaluation of the utilizations a negotiation protocol has to be used. It is not decided yet if this will be the contract net (see [18]) or a more sophisticated approach. Since our self-controlled agv system should be applicable in a real scenario the message volume per time unit has to be considered since the messages have to be transported wireless for example with wlan.

2.3. Order Allocation

As Boucké et.al. stated out dynamic order allocation is needed if changes can arise between the initial allocation of an order and the start of the corresponding transportation task (see subsection 1.2 and [20]). Beside unexpected things that can change the circumstances under which an order allocation had taken place like the break down of an agv or the restart of an so far broken down agv in our approach these circumstances change constantly. The reason for this is that the agvs can loose reservations through negotiations which can lead to longer routes.

Two approaches will be investigated. First we will try to install the algorithm presented by Boucké et.al. This approach is tested and have shown good results but it also uses a significant amount of messages.

The second approach will be the use of contract nets with the added possibility that agvs can try to give orders back. This means that an agv will cyclic check if one of the following condition holds:

$$\frac{t_{orderEndAtAllocation}}{t_{orderEndCurrent}} > \delta \quad (2)$$

$$\text{time since last auction} > \tau \tag{3}$$

If that is the case the agv acts as an task-manager and start a special kind of auction in which he only accepts proposals which are better than its own (or to be exact the other agvs are only sending proposal that are better). With this approach we hope that on the one hand auctions can be reassigned if such a reassignment would improve the result significantly (where "significantly" can be set with the parameter δ) but on the other hand do not lead to a significant extra computation time or an extra amount of messages.

3. Main Challenges

The main challenges that must be passed to successfully implement our approaches as described in the previous section are:

1. Find and order significant edges prior to negotiations.
2. Determination of good parameters.
3. Restriction of the needed amount of messages.

To solve challenge 1 we want to develop an algorithm that evaluates the reservations by considering the detour they cause as well the likelihood that they are withdrawn and that not only for single reservations but also for sets of reservations. For example it may be more useful for an agv to try to negotiate about a set of reservations which are owned only by a fey other vehicles than to negotiate about a set which are owned by many different agvs even if it would lead to a slightly shorter route.

In the approaches described in section 2 some parameters occur. For example ε which determines if negotiations should be started, max response time whereby the time for negotions is limited, the parameters of the scoring function (p_1, p_2, p_3 - see Eq. 1), δ which describes how much delay there has to be before an agv tries to reallocate an order which was assigned to it or τ which is the time after which an agv tries to reallocate orders. During the implementation and design phase more parameters will follow (more influences on the scoring function and thus more parameters, parameters which control how the significant reservations are determined, and so on.). In order to determine good sets of parameters and thus passing challenge 2 we want to use heuristics, genetic algorithms (see for example [21]), simulated annealing (see for example [22]), tabu search (see for example [23]), ant colony optimization (see [24]) or similar approaches. Here some further questions arise: Should all parameters be determined as one set or sould they be splitted? Is it useful to determine scenario specific parameters? Should the system be a learning one which can change the parameters by its own?

Since we want to develop an agent system which is applicable to real scenarios the communications bandwidth (for example the wlan bandwidth) will be limited. Thus we have to design the needed negotiation and allocation protocols in a way that they don not overload these bandwidth. At the moment it is not clear how hard this challenge number 3 will be.

4. Planned Evaluation

After designing and developing the concepts and algorithms described in section 2 they should be evaluated in three ways. First we want to test the algorithms in abstract graphs

to compare the results in terms of running time and quality of the solutions with other algorithms like the ones presented in section 1.

Than we want to test the hole self-controlled agv system in realistic scenarios. For this purpose we have access to two scenarios in which agv system were installed through one of the industry partners of the FTS-Selbststeuerung research project. One scenario is a hospital with several floors and elevators connecting them the other is a beverage bottling factory. In these evaluations we want to test if the solutions work as planned when disturbances like delayed or lost messages occur. Furthermore we want to compare our results with the ones of the installed system in terms of delivery reliability, throughput, delays and utilization of the agvs.

In the third stage of evaluation we want to investigate how robust the system is to common kinds of errors and failures like lost messages, blocked way segments, break down of vehicles, and so on. We furthermore want to investigate how installation and adjustment costs behave compared to centralized agv systems as they are built by our industry partners nowadays.

5. Conclusion

The research in the field of self-controlled agv systems in the last years have showed that good decentralized algorithms for the main problems like conflict-free routing and decentralized order allocation exits. But until now negotiations are not used at the route planing or at least not as an integral part.

In this paper we presented our planned approach to add negotiations into self-controlled agv systems. We also point out main challenges that arise and describe how they can be solved.

If our approach is successful and the evaluations show a real benefit from adding negotiations further research questions can be asked. For example we could allow vehicles to split orders. This means a vehicle can apply for an transport task which should move a transport unit from a to c even if it only wants to transport it from a to b (for example because it already has accepted an order which starts at b) if it finds another vehicle which would transport it from b to c. Or we can investigate scenarios in which agvs and continuous conveyors are installed and the agvs can use the continuous conveyors. Of course this questions arise only for scenarios that allow such order splittings.

Acknowledgements

This research is done as part of the FTS-Selbststeuerung (agv self-control) project. The project is funded from the budget of the German Federal Ministry of Economics and Technology (BMWi) via the Federation of Industrial Cooperative Research Associations "Otto von Guericke" (AIF) for the Federal Logistics Association (BVL).

References

[1] Guido Follert and Moritz Roidl. Evaluation of Routing Strategies for Decentralized Self-Organisation in Large Scale Conveyor Systems. *Kimberly E. et al.(Hrsg. Bd.): Progress*, pages 1–25, 2008.

[2] James A. Tompkins, John A. White, Yavuz A. Bozer, and J.M.A. Tanchoco. *Facilities Planning*. Wiley, 4 edition, 1 2010.

[3] A. Schuldt. Decentralisation and Interaction Efficiency in Cooperating Autonomous Logistics Processes. In H.-J. Kreowski, B. Scholz-Reiter, and K.-D. Thoben, editors, *2nd International Conference on Dynamics in Logistics (LDIC 2009)*, pages 269–278, Bremen, Germany, 2009. Springer-Verlag.

[4] Cornelis Versteegt. *Holonic Control For Large Scale Automated Logistic Systems*. PhD thesis, 2004.

[5] Sebastian Schreiber and Alexander Fay. Requirements for the benchmarking of decentralized manufacturing control systems. *2011 IEEE 16th Conference on Emerging Technologies Factory Automation (ETFA)*, 2011.

[6] B. Scholz-Reiter, J. Kolditz, and T. Hildebrandt. Uml as a basis to model autonomous production systems. *Proceedings of the 3rd CIRP Sponsored Conference on Digital Enterprise Technology*, pages 1–8, 2006.

[7] Namita Singh, P. V. Sarngadharan, and Prabir K. Pal. AGV scheduling for automated material distribution: a case study. *Journal of Intelligent Manufacturing*, 22(2):219–228, July 2009.

[8] FIPA. *FIPA Contract Net Interaction Protocol Specification*. FIPA, 2001.

[9] A.J. Broadbent, C.B. Besant, S.K. Premi, and S.P. Walker. Free ranging agv systems: promises, problems and pathways. *Proc. 2nd Int. Conf. on Automated Materials Handling*, pages 221–237, 1985.

[10] E.W. Dijkstra. A note on two problems in connecting with graphs. *Numerische Mathematik*, 1:269 – 271, 1959.

[11] C.W. Kim and J.M.A. Tanchoco. Conflict-free shortest-time bidirectional AGV routeing. *International Journal of Production Research*, 29(12):2377–2391, 1991.

[12] R.H. Möhring, E. Köhler, E. Gawrilow, and B. Stenzel. Conflict-free real-time AGV routing, 2004.

[13] N.G.F. Sancho. Shortest path problems with time windows on nodes and arcs. *Journal of Mathematical Analysis and Applications*, 186(3):643 – 648, 1994.

[14] AW ter Mors and J. Zutt. Context-aware logistic routing and scheduling. *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling*, pages 328–335, 2007.

[15] Nenad Smolic-Rocak, Stjepan Bogdan, Kovacic Zdenko, and Tamara Petrovic. Time windows based dynamic routing in multi-AGV systems. *IEEE Transactions on Automation Science and Engeneering*, 7(1):151–155, 2010.

[16] Jacques Ferber. *Multi-agent systems : an introduction to distributed artificial intelligence*. Addison-Wesley, Harlow, 1998.

[17] N.R. Jennings, P. Faratin, A.R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge. Automated negotiation: prospects, methods and challenges. *Int. Journal of Group Decision and Negotiation*, pages 1–30, 2001.

[18] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Trans. Comput.*, 29(12):1104–1113, 1980.

[19] Hadeli, Paul Valckenaers, Martin Kollingbaum, and Hendrik Van Brussel. Multi-agent coordination and control using stigmergy. *Computers in Industry*, 53(1):75 – 96, 2004.

[20] Nelis Boucké, Danny Weyns, Tom Holvoet, and Koenraad Mertens. Decentralized Allocation of Tasks with Delayed Commencement. *In Proceedings of European Workshop on Multiagent Systems*, 2004.

[21] Sean Luke. *Essentials of Metaheuristics*. Lulu, 2011. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/.

[22] Ingo Wegener. Simulated annealing beats metropolis in combinatorial optimization. In *Automata, Languages and Programming*, volume 3580 of *Lecture Notes in Computer Science*, pages 61–61. Springer Berlin / Heidelberg, 2005.

[23] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[24] Marco Dorigo and Mauro Birattari. Ant colony optimization. In *Encyclopedia of Machine Learning*, pages 36–39. 2010.