2004 IEEE International Conference on Systems, Man and Cybernetics

Recognizing and Handling the Livelock Problem in AGV Systems*

Elzbieta Roszkowska
Institute of Engineering Cybernetics, Wroclaw University of Technology, Poland
ekr@pwr.wroc.pl

Abstract – *The paper concerns AGV systems (Automated Guided Vehicle Systems or AGVSs) with bidirectional guidepath network, unidirectional vehicles, zone control for avoiding collisions, and dynamic route planning. An AGVS is represented with a DES that models the feasible system dynamics. We study the liveness problem of the system and associate this property with the permanent ability of the vehicles to attain any edge in the network. The paper is a continuation of our earlier work, where we established necessary and sufficient conditions for a state of AGVS to be live, for each possible case except a singularity case, where (unlike in the other cases) the occurrence of livelocks is possible. We define the notion of a livelock with respect to the AGVS model and conduct the analysis that results in a necessary and sufficient condition for the occurence of this phenomenon.*

Keywords: AGV system, DES, deadlock, livelock.

1 Introduction

Automatic Guided Vehicle Systems (AGV systems or AGVSs) are one of the most exciting and dynamic areas in material handling today. Typically, AGVs have been used in applications involving non-continuous flow of materials in flexible manufacturing systems (FMSs). Examples include delivering components to workstations, transferring work-in-process parts in and out of buffers, or transporting products to be assembled from one station to another throughout the assembly process. Other applications of AGVs include automated container seaport terminals, e.g. [2], where a big number of vehicles, operating in a relatively small area, look after the transport of containers between the quays and the container store places. In recent years a large research project was also devoted to the analysis of the possibilities for underground logistic systems that use AGVs to carry goods on a large scale [10]. Such systems are aimed at, e.g., city distribution with logistical centers at the edges of the city, cargo distribution in industrial zones, or a replacement for traditional transport on high-volume transport links.

Until recently, the problem of the coordination of vehicle movement in AGVS was almost exclusively solved through eliminating conflicts among vehicles by a special design of the network, such as the tandem configuration, simple unidirectional networks, e.g. [1]), or eliminating vehicle conflicts during the vehicle route planning and scheduling stage, e.g. [4]. The former approach has many practical implementations, although both analytical and experimental results, e.g. [4], indicate higher efficiency and lower cost of bidirectional networks. The major shortcoming of the schedule-based approach to the vehicle coordination is its open-loop control character. Such policies are non-robust and very sensitive to the system randomness. A more recently evolving approach is to handle vehicle conflict in real-time by modeling the AGVS as a DES (Discrete Event System) [3, 5–9, 11]. The contributions can be classified with respect to certain features of the AGV model such as: the type of the system (open vs. closed systems), the routing scheme (pre-determined vs. dynamically established routes), the modeling formalism (Petri nets, deterministic automata, processes/resources OS-like representation), and the research tendency (analysis-oriented vs. synthesis-oriented).

The paper is a continuation of our earlier work [7, 8], where we established necessary and sufficient conditions for a state of AGVS to be live, for each possible case except a singularity case, i.e. the system with $k = b + 1$ free zones, where b is the number of bridge edges in the graph abstracting the guide path network. In particular, for biconnected graphs, where $b = 0$, this corresponds to networks with only one free zone. The singularity of the case $k = b + 1$ stems from the livelock phenomenon, whose occurence is only possible here. We define the notion of a livelock with respect to the AGVS model and conduct the analysis that results in a necessary and sufficient condition for the state liveness in this class of systems.

2 The model

We will consider the class of AGV systems with bidirectional guidepath networks, unidirectional vehicles, zone control for avoiding physical collisions, and dynamic vehicle route planning. It is assumed that the structure of the guidepath network is given by a a graph, whose edges correspond to sections, or zones, and vertices to intersections and zone interconnecting points. A path segment can be traveled in both directions, yet once a vehicle has entered a zone through

*0-7803-8566-7/04/$20.00

one end, it can only proceed toward the other end. The zone control refers to that no more than one vehicle is allowed to occupy a zone at a time, and the dynamic route planning implies that the detailed path that a vehicle will follow when accomplishing a mission is determined on-line. Having arrived at the end of a section, a vehicle can be commanded to enter any unoccupied neighboring section, or to stop and wait until it is ordered to proceed.

The operation of such a system can be viewed as a DES, whose state is given by the state of the zones (empty or occupied by a vehicle moving in one of the two directions), and in which a state transition occurs when a vehicle located in one zone moves to another.

Definition 1 A vehicle system is a five-tuple $VS = (U, S, T, \varphi, \delta)$, where:

1. $U = (V, E, in)$ is an undirected graph such that each vertex $v \in V$ is incident to at least two edges, called *path graph*; the edge set E corresponds to the network zones, the vertex set V to the zone interconnecting points, and $in : E \to \{\{v, v'\}, \{v\} \subseteq V\}$ is the incidence function.

2. S is the state space of VS, where each state $s \in S$ is given by a function $s : E \to V \cup \{null\}$ s.t. $s(e) = v \Rightarrow v \in in(e)$, called vehicle configuration. $s(e)$ describes the state of the zone corresponding to edge e (or of zone e, for short). If $s(e) = v$ then there is a vehicle in zone e moving towards vertex v. If $s(e) = null$ then zone e is empty.

3. $T = \{t = (e_t, e'_t) \mid in(e) \cap in(e') \neq \emptyset\}$ is the set of system events corresponding to the vehicle location change from zone e_t to e'_t.

4. $\varphi : S \times T \to \{enabled, disabled\}$ is *feasibility function*. For each pair (s, t), $\varphi(s, t)$ determines whether at state s event $t = (e_t, e'_t)$ can occur. $\varphi(s, t) = enabled$ iff $s(e_t) \in in(e'_t)$ and $s(e'_t) = null$.

5. $\delta : S \times T \to S$ is a partial function defined for each pair (s, t) such that event $t = (e_t, e'_t)$ is enabled at state s, called *next-state function*. For each edge $e \in E$, the new state $s'(e)$ is given by

$$s'(e) = \begin{cases} null & \text{if } e = e_t \\ v'_t & \text{if } e = e'_t \\ s(e) & \text{otherwise} \end{cases}$$

where $\{v'_t\} = in(e'_t) - \{s(e_t)\}$, if e'_t is not a self-loop, and $\{v'_t\} = v_t$ otherwise.

The system defined above has a convenient graphical representation (see Fig. 1 for an example). It can be viewed as a partially directed graph (or pd-graph) $G(s) = (U, s)$, built on the undirected graph U, that represents the guide path network, and described with the variable edge direction function $s(e)$. The direction attribute of an edge abstracts

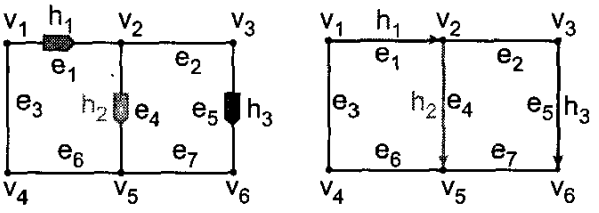

Figure 1: An example AGV guidepath network U at state $s = [v2, null, null, v5, v6, null, null]$ and the corresponding partially directed graph $G(s) = (U, s)$.

the state of the corresponding zone. An undirected edge represents an empty zone, while a directed edge represents a zone with a vehicle moving in the direction of the edge $s(e)$. The event set T consists of all pairs $t = (e, e')$ of edges that have a common vertex, and abstracts vehicle location changes. Event t is enabled at state s, that is, a vehicle can move from zone e to e', if edge e' is undirected, and edge e is directed and points to vertex v shared with e'. The occurrence of t changes the state of the network from s to s' so that edge e becomes undirected, and edge e' becomes directed and points from v.

Additionally, if needed to observe the movement of distinguished vehicles, we will assume that the directed edges are labeled uniquely with elements h of the vehicle set H. If vehicle h moves from one edge to another then the label moves along with the vehicle. Formally, the labels h are given by function $\gamma(s, e)$ defined as follows.

Definition 2 Let $s_0 \in S$ be any distinguished state of a vehicle system VS. The edge label function $\gamma : S \times E \to H \cup \{null\}$, where H is a set of vehicle symbols, is defined inductively as follows.

1. $\gamma(s_0, e)$ is any function such that: (i) $\gamma(s_0, e) = null$ iff $s_0(e) = null$, and (ii) for each two distinct edges $e, e' \in E$ s.t. $s_0(e) \neq null$, $\gamma(s_0, e) \neq \gamma(s_0, e')$.

2. for each state transition $s' = \delta(s, t)$, $t = (e_t, e'_t)$,

$$\gamma(s', e) = \begin{cases} null & \text{if } e = e_t \\ \gamma(s, e_t) & \text{if } e = e'_t \\ \gamma(s, e) & \text{otherwise} \end{cases}$$

For each state $s \in S$, its reachability set, i.e. the set of states that consists of state s and all states reachable directly and indirectly from s, will be denoted with $R(s)$.

Definition 3 If an event $t \in T$ is feasible at state $s \in S$, then the fact that a new state s' can be reached is denoted by $s\,[t>\, s'$. The reachability set $R(s)$ is the set of states, such that (i) $s \in R(s)$, and (ii) if $s' \in R(s)$ and $s'\,[t>\, s''$ then $s'' \in R(s)$.

3 State liveness analysis

Unlike in an open AGV system, where having finished its mission a vehicle leaves the guidepath network, in a closed

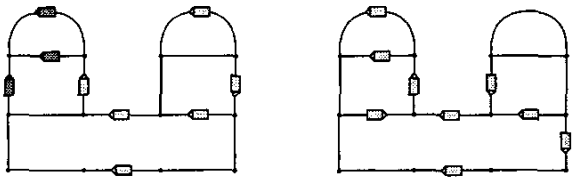

Figure 2: A deadlock and an impending deadlock.

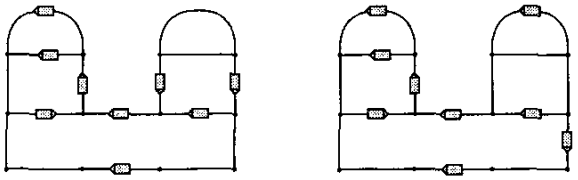

Figure 3: Live states.

AGVS the vehicles must be able to accomplish any task and remain operational, that is have the constant potential to visit any given destination point in the guidepath network. The development of supervisory control that enforces such a behavior requires formal tools that allow to distinguish between a state that supports this ability, and a state that deprives the vehicles of this ability. The former states will be called *live* and defined as follows.

Definition 4 Let $s_0 \in S$ be any distinguished state of a vehicle system VS. State $s \in R(s_0)$ is live iff: (i) for each pair of edges $(e, e') \in E \times E$ such that $\gamma(e, s) = h \in H$, there exists state $s' \in R(s)$ such that $\gamma(e', s') = h$, and (ii) state s' is live.

Less formally, state $s \in S$ is live if and only if each vehicle h, say located in zone e, can attain any given zone, say e', in the network, and the resulting state is live. The phenomena that can deprive the state from its liveness property are deadlocks and livelocks.

Definition 5 In system VS at state s occurs: a *deadlock*, if there exists a vehicle that cannot leave its zone at any state $s' \in R(s)$, and a *livelock*, if each vehicle can move infinitely in a particular subnetwork, but leaving the subnetwork is either infeasible or results in a deadlock.

It is clear that when a deadlock occurs (see Fig. 2 (left) for an example), the state is not live, as those vehicles that deadlock cannot move any further, while the others can never reach the zones occupied by the former. Fig. 2 (right) shows

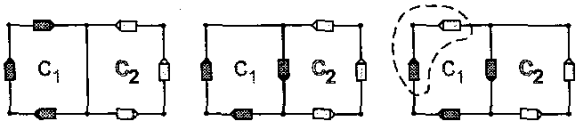

Figure 4: A livelock configuration.

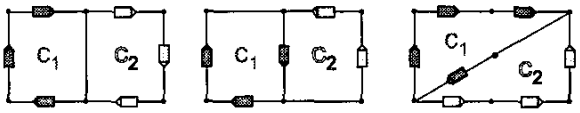

Figure 5: Live states.

a state which is not live either, because any further movement of the vehicles unavoidably leads to a deadlock. As opposite to those, both states in an example AGVS, depicted in Fig. 3, are live. Fig. 4 (left and middle) gives an example of a livelock. The vehicles located on cycle c_1 can circulate infinitely on this cycle, and the vehicles located on cycle c_2 can circulate infinitely on that cycle, yet none of them can move from one cycle to the other without causing a deadlock (Fig. 4, right). On the other hand, similar situations but in a system with a different cycle structure (Fig. 5, left), with one less vehicle (middle), or with one more edge shared by the cycles (right), are examples of live states. It is not hard to notice that now the vehicles can circulate on the cycles that they are initially located on, as well as move from one cycle to the other.

In more complex cases than those presented above, based solely on Def. 4, it is practically impossible to determine whether or not a particular state is live. To solve this problem, it is necessary to reduce its complexity by establishing equivalent, yet easier to verify liveness conditions. Such conditions have been established in [7, 8] and are shortly recaptured below. We assume that the reader is familiar with the basic terms concerning graphs. Therefore we will only refresh some less commonly used standard notions, and recall briefly the notions defined for the new graph type - the partially directed graph.

Definition 6 For an undirected graph U: Vertex v is an *articulation point* if there exist vertices v' and v'' such that v, v', and v'' are distinct and each path between v' and v'' contains v. A bridge is a maximal path such that each vertex on the path is an articulation point. Graph U is *biconnected* if it is connected and has no articulation points. A biconnected component of U is a maximally biconnected subgraph of U.

Definition 7 For a pd-graph $G = (U, s)$: A sequence of edges π is a *path* in G if π is a path in U and each directed edge e on π is followed from tail to head of e. Path π is *simple* if all vertices on the path, except possibly the first and last vertices, are distinct. A simple path with the same first and last vertex is a *cycle*.

Definition 8 In a pd-graph $G = (U, s)$:

1. A *joint* between two cycles c and c' is a simple path π that lies on both cycles and contains all edges shared by c and c'. A *pass* between two cycles c and c' is a simple path π, whose all edges are undirected, and one of the end vertices of π lies on c, the other on c'.

2. A *chain* is a subgraph of G that has the form of a sequence $ch = c_1, \pi_1, c_2, \pi_2, \ldots, \pi_{n-1}, c_n$ such that for

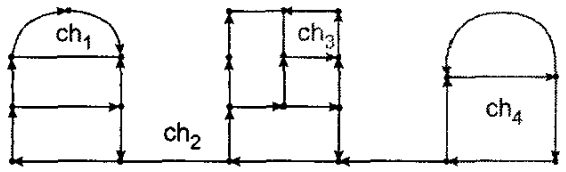

Figure 6: Chains in a partially directed graph.

each $i \in 1..n-1$, path π_i is either a joint or a pass between cycles c_i and c_{i+1} (see Fig 6 for an example).

3. Two edges $e, e' \in E$ are *chain-connected* or *chained* iff there exists a chain that contains both e and e'. Graph G is chained iff any two arbitrary edges $e, e' \in E$ are chained.

A chained pd-graph has an important property.

Property 1 A pd-graph $G = (U, s)$ is chained iff each bridge edge is undirected and each non-bridge edge lies on a cycle.

Table 1: Classification of systems VS with respect to the number of bridge edges b and free zones k.

number of free zones	state liveness conditions $\forall s \in S$
$k > b+1$	state s is live $\Leftrightarrow$ $\exists s' \in R(s)$ s.t. $G' = (U, s')$ is chained
$k = b+1$	state s is live $\Rightarrow$ $\exists s' \in R(s)$ s.t. $G' = (U, s')$ is chained state s is live $\Leftrightarrow$?
$k \leq b$	state s is not live

It follows that in order to complete the liveness analysis, we need to establish liveness conditions for the singularity case – systems VS with $k = b + 1$ free zones. In the sequel, we will refer to these systems as class $\mathcal{VS}2$, whereas the systems with $k > b + 1$ and with $k \leq b$ will constitute classes $\mathcal{VS}1$ and $\mathcal{VS}3$, respectively. For the sake of convenience, instead of the long term 'state s such that graph $G(s)$ is chained', we will use the short term 'chained state s'.

4 State liveness in class $\mathcal{VS}2$

A basic step in the development of the correct control for an AGVS is to ensure that it starts operation at a live state. In the open vehicle system, it is possible to distinguish in a natural way a particular live state - the state when no vehicles are present in the network. In the closed vehicle system such an assumption is not possible, yet, based on the results presented in Table 1, it seems reasonable to assume that the initial state s of VS should be chained. Such an assumption ensures that each system $VS \in \mathcal{VS}1$ is live at s, and that for each system $VS \in \mathcal{VS}2$, an important necessary condition for the liveness of s holds. Let us concentrate on the latter class of AGV systems.

Property 2 Let s be any chained state in system $VS \in \mathcal{VS}2$. At state s there is no vehicle on any bridge edge and there is exactly one biconnected component BC_i, $i \in 1..m$, with an empty zone, whereas all other zones in this and all other biconnected components are occupied with vehicles.

Proof. Since in this class there are $k = b + 1$ empty zones at any state s (or, equivalently, $k = b + 1$ undirected edges in graph $G(s)$) and, by Property 1, each bridge edge is undirected at a chained state, the thesis is evident. Fig. 7 gives an example of such a system. ■

Similar to the abstraction used in the solid state theory, where one observes both the flows of electrons and of holes, it is convenient to follow the movement of both vehicles and 'holes' in AGVS. The holes move along with the vehicles, but in the opposite direction, that is, when a vehicle moves from edge e to edge e', a hole moves from edge e' to e. The holes' dynamics obey the following property.

Property 3 Let s be any chained state in system $VS \in \mathcal{VS}2$. For each zone $e \in E$, there exists a chained state $s' \in R(s)$ such that zone e contains a hole.

Outline of proof. Since $G(s)$ is chained then any two given edges $e, e' \in E$ lie on a chain. By Property 2, there exists a chain with a hole on a cycle. The hole can circulate on this cycle as well as move to a neighbor cycle without destroying the structure of the chains. Thus the hole can attain any edge e in the network so that at the resulting state s' graph $G(s')$ is chained. ■

Property 4 Let s be any chained state in system $VS \in \mathcal{VS}2$. For each state $s' \in R(s)$, there is no chained state $s'' \in R(s')$ if at state s' there exists a biconnected component BC_i, $i = 1..m$, with more than one hole.

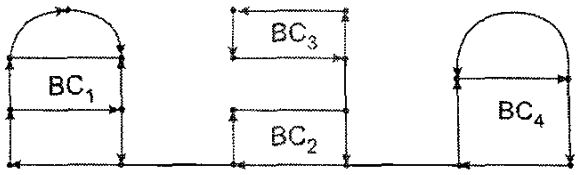

Figure 7: A system $VS \in \mathcal{VS}2$ at a chained state s.

Outline of proof. Clearly, each vehicle leaving a component for a bridge will eventually deadlock unless it can enter another component. Thus, in order not to deadlock, a vehicle can only leave its component, say BC_1, for a bridge edge that starts a path to a component with a hole, say component BC_2. Since, by Property 2, at state s, there is only one such a component, if a vehicle leaves BC_1 then a deadlock occurs. Since BC_1 contains only one empty zone, each next vehicle that leaves BC_2 will cause a deadlock, because it can neither be accomodated in BC_1 (as the first vehicle will reach it first) nor in any component B_i, $i = 3..m$ (as there is no empty zone), nor again in BC_2 (as the vehicle cannot return until it enters a biconnected component). The same can be demonstrated for any other component B_i, $i = 3..m$, which proves the thesis. ■

Property 5 Let s be any chained state in system $VS \in \mathcal{VS}2$. At each live state $s' \in R(s)$, each pd-graph $G_i(s') = (BC_i, s')$, $i = 1..m$, is chained.

Outline of proof. At state s, all subgraphs $G_i(s)$, $i = 1..m$, are chained. Consider a state transition $s''[t > s'$, $s'' \in R(s)$ and assume a that for a particular component BC_i, graph $G_i(s'')$ is chained and graph $G_i(s')$ is not. The state transition from s'' to s' is associated with a partition of the chained graph $G_i(s')$ into two chained subgraphs connected with two separate paths going in the same direction If there is more than one hole in $G_i(s')$ then, by Property 4, there is no chained state $s'' \in R(s')$, hence s' is not live. If there is only one hole in $G_i(s'')$ then the vehicle that causes the transition leaves an empty zone behind it, and at state s' occurs a deadlock. Thus s' is not live. ■

Note now that any system V can be viewed as an ensemble of systems VS_i, $i = 1..m$, associated with the respective biconnected components BC_i. In order to state the final theorems, let us define the following.

Definition 9 Let VS_i, $i = 1..m$, be the vehicle systems associated with the biconnected components of system $VS \in \mathcal{VS}2$. For state s in system VS, s_i is a state of system VS_i such that: a) $s_i(e) = s(e) = null$, if there exists an empty zone in VS_i, and $s_i(e) = null$ for an arbitrary zone e in VS_i, otherwise, and b) $s_i(e') = s(e')$ for each other zone $e' \neq e$ in VS_i.

Property 6 In system $VS \in \mathcal{VS}2$, a chained state s is live if and only if for each $i = 1..m$, state s_i in system VS_i is live.

Outline of proof. By Property 3 a hole can be transmitted to any zone in the system so that a chained state is preserved. Thus, in each system VS_i state s_i is reachable from any chained state. If s_i is live then each vehicle in VS_i can attain each zone in the component, and or/move to an adjacent component, as at a chained state all bridge zones are empty. Consequently, if each s_i is live then state s is live. If any of the states s_i is not live then state s is not live either, because some zones in VS_i are not attainable with the current chain structure, and with a single hole (by Property 4, a necessary condition for state liveness) this structure cannot be changed. ■

Theorem 1 Let s be any chained state in system $VS \in \mathcal{VS}2$. For each $s' \in R(s)$, state $s' \in R(s)$ is live if and only if a) state s is live, and b) there exists a chained state $s'' \in R(s')$.

Outline of proof. $\Rightarrow$ If state s' is live then condition (a) is obvious, and condition (b) is necessary by Table 1. $\Leftarrow$ If there exists a chained state $s'' \in R(s')$ then, by Property 4, at each state in any execution sequence that leads to s'', no biconnected component has more than one hole. Thus, each $G_i(s'')$ preserves the chain structure of $G_i(s)$. If state s is live then, by Property 6, for each $i = 1..m$, state s_i in system VS_i is live. Consequently, state s'' is live. Since $s'' \in R(s')$, state s' is live. ■

Theorem 1 proves that the liveness conditions for systems $VS \in \mathcal{VS}2$ and a live and chained initial state are the same as for systems $VS \in \mathcal{VS}1$. Thus, the same control can be applied for both classes, providing hat we can determine whether or not a particular chained system $VS \in \mathcal{VS}2$ is live.

The presented result concerns systems VS whose initial state is chained. In the general case we obtain the following.

Theorem 2 In system $VS \in \mathcal{VS}2$, state $s \in S$ is live if and only if there exists state $s' \in R(s)$ s.t. s' is chained and for each $i = 1..m$, state s'_i of system VS_i is live.

Outline of proof. Implied directly by Theorem 1. ■

5 Livelock detection

A practical application of the results stated by Theorems 1-2 requires a method to determine whether or not state s_i in each system VS_i is live. By Def. 9 and Properties 4-5, it is equivalent to determining whether or not a chained state s in system VS with a biconnected network U and only one empty zone is live. Since state s is chained then in $G(s)$ each vehicle is located on a cycle and there is a chain that contains any two cycles in the network. As, by Property 3, the hole can attain any edge in the network, then all vehicles can circulate infinitely on the cycles, so no deadlock or an impending deadlock occurs. Consequently, that state s is not live iff a vehicle cannot move from one cycle to a neighbour cycle, that is, occurs a livelock. The following definition formally describes a graph structure that, as we will see, assists such a phenomenon (see also Fig.8).

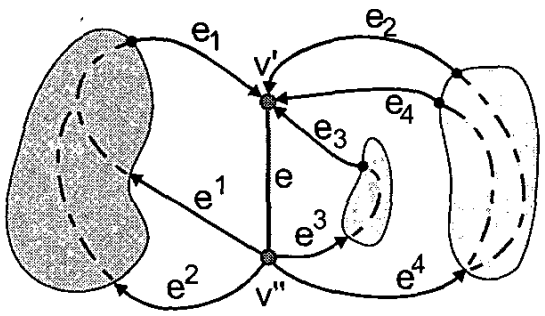

Figure 8: A bar edge. Illustration to Definition 10.

Definition 10 In a pd-graph G, edge e with distinct vertices v' and v'' is a *bar edge* iff it has the following properties: a) vertex v obtained by merging v' and v'' into one vertex is an articulation point in the new graph resulting from the merging operation, b) each edge $e_i \neq e$ incident to v' lies on a cycle that includes path $\pi_i = w_i e_i v' e\, v''$, and π_i is the only path in G from w_i to v'', c) each edge $e^i \neq e$ incident to v'' lies on a cycle that includes path $\pi^i = v' e\, v'' e^i w^i$, and π^i is the only path in G from v' to w''.

Theorem 3 In system VS a chained state s is live if and only if: a) $VS \in \mathcal{VS}1$ or b1) $VS \in \mathcal{VS}2$ and b2) there is no bar edge in graph $G(s)$.

Outline of proof. $\Rightarrow$ By earlier results (Table 1) it is clear that $VS \notin \mathcal{VS}3$, and that in the case of $VS \in \mathcal{VS}1$, condition (a) is sufficient for the liveness of s. If $VS \in \mathcal{VS}2$ then there is only one hole in the network. When a vehicle enters the bar edge, the hole allows the vehicles to circulate in their respective subgraphs (depicted with different colors in Fig. 8), but not move from one subgraph to another without causing a deadlock. Thus, if state s is live then there is no bar edge in $G(s)$. $\Leftarrow$ If $VS \in \mathcal{VS}1$ then a chained state is live. If $VS \in \mathcal{VS}2$ and there is no bar edge in graph $G(s)$ then it can be demonstrated that no edge e satisfies all the condition given in Def. 10. If any of them is missing (as e.g. in Fig. 5) then any vehicle can move between cycles that share edge e. ■

6 Conclusion

The presented paper concerns the state liveness problem in closed AGV systems with dynamic vehicle routing. We distinguish three clases of such systems: $\mathcal{VS}1$ - $\mathcal{VS}3$, and concentrate on $\mathcal{VS}2$ – the singularity case, where the occurence of livelocks is possible. As the remaining classes have already been examined in our erlier works, this completes a certain stage of the vehicles' dynamics analysis. Our major contribution is the theorem proving that no livelock can ever occur once the initial state is live and chained, and the necessary and sufficient conditions for a livelock occurence in a chained state. This allows us to reduce the livelock avoidance problem to the livelock detection, that can be done at the system design phase.

Acknowledgment. This work was supported by the Ministry of Scientifi c Research and Information Technology, Poland, under grant no. 5 T12C 02625.

References

[1] Y. Bozer and M. Srinivasan. Tandem configurations for AGV systems and the analysis of single vehicle loops. *IEE Trans.*, 23:72–82, 1991.

[2] M.B. Duinkerken, J.M. Evers, and J.A. Ottjes. A generic simulation model for systems container terminals. In *Proceedings of the 16th European Simulation Multiconference ESM2002*, Darmstadt, 2002.

[3] M. P. Fanti. Event-based controller to avoid deadlock and collisions in zone-control AGVS. *International Journal of Production Research*, 40(6):1453–1478, 2002.

[4] N. Krishnamurthy, R. Batta, and M. Karwan. Developing conflict free routes for automated guided vehicles. *Operations Research*, 41:1077–1090, 1993.

[5] S. A. Reveliotis. Conflict resolution in AGV systems. *IIE Transactions*, 32(7):647–659, 2000.

[6] E. Roszkowska. Joint approach to design and control of process flows to avoid deadlocks in flexible production systems. In *Intelligent Systems in Design and Manufacturing*, volume SPIE 3517, pages 334–345, 1998.

[7] E. Roszkowska. Liveness of states in closed AGV systems with dynamic routing. In *IEEE Eighth Int. Symp. on Methods and Models in Automation and Robotics*, Szczecin, Poland, September 2002.

[8] E. Roszkowska. Undirected colored Petri net for modelling and supervisory control of AGV systems. In *6-th Int. Workshop on Discrete Event Systems WODES'02*, pages 135–142, Zaragoza, Spain, October 2002.

[9] E. Roszkowska. Supervisory control for closed AGV systems with dynamic vehicle routing. In *7th Int. IFAC Symp. on Robot Control SYROCO'2003*, Wroclaw, Poland, September 2003.

[10] A. Verbraeck, E. Valentin, and Y.A. Saanen. Simulation as a real-time logistic control system: AGV control with simple++. In *The New Simulation in Production and Logistics Prospects, Views and Attitudes*, pages 245–255, 2000.

[11] N. Wu and M. Zhou. AGV routing for conflict resolution in AGV systems. In *2003 IEEE Int. Conf. Robotics & Automation*, pages 1428–1433. Taipei, Taiwan, September 2003.