INTERNATIONAL JOURNAL OF INTELLIGENT CONTROL AND SYSTEMS
VOL. 12, NO. 1, MARCH 2007, 82-94

Control of Automatic Guided Vehicles in the Manufacturing Systems Using Petri Nets

Branislav HRÚZ

Abstract- **An analysis of the transportation systems used in flexible manufacturing systems is introduced in the paper. Based on the analysis an approach to the modelling and control of an automatic guided vehicles system (AGVS) is presented. It is aimed at a specifically delimited class of the AGV transportation systems. A special group of the colored Petri nets interpreted for control is used for the process control specification. A time optimal solution of the AGV route planning is proposed.**

***Index Terms*—Discrete event dynamic systems, AGV systems, Petri nets, colored Petri nets, optimization.**

1. INTRODUCTION

Processing objects and their transfers between nodes in discrete event systems represent important problems of the system theory and engineering practice. Their efficient solutions need adequate models and control methods.

Flexible manufacturing systems (FMS) are a class of discrete event systems where the above-mentioned problems are to be often and urgently solved. Automatic guided vehicles (AGV) are important transportation means in FMS, especially when used for the mass production in automobile, aircraft and similar industries. There AGV systems represent subsystems consisting of vehicles or carts providing the transport of palettes, work-pieces, fixtures, etc., among machining cells, processing centers, robots, measuring devices, buffers, and storages. The transport system performance combined with the production job scheduling determines the overall efficiency of FMS [1].

An overall approach to the optimal control of FMS includes the following three aspects of analysis and design to be mutually dependent:

1) design of FMS and scheduling of object processing (processing is realized by operations);
2) structuring of the AGV transportation subsystem; and
3) scheduling and routing of AGVs in the transportation subsystem.

Seo and Egbelu [2] treated all three mentioned aspects and proposed a global solution method respecting mutual relations and influences. A problem in practical applications is lack of adequate data required for the overall approach and relatively complicated calculations impossible to be executed in real-time conditions including failures of the system components.

The consequence and a way out of the described difficulties is that designers of FMS proceed rather in heuristic ways and they solve the above mentioned problems separately and independently. After the FMS design suitable for the required product class, they usually propose a transportation structure matched and adapted to the particular processing requirements. Consequently, the transportation flexibility is limited and the optimal guiding of AGV is restricted just for certain specific production tasks. Operation scheduling is solved separately from AGV scheduling. Ling and Weng-Jing [3] reported and studied many known modeling and control methods of the automatic guided vehicle system (AGVS) and showed that a universal method to be applied in any practical application had not yet been available and was questionable.

This paper is devoted to the modeling and control of AGVS assuming that the transportation system structure or configuration is given as well as transportation tasks resulting from the operation scheduling or transportation tasks appearing spontaneously. The transportation system structure should be designed in a specific way as it will be described later.

The autonomy level of vehicles and their "ability of behaving intelligently" can be different. A higher level assumes that vehicles have actual information of positions, speeds, movement directions, and executed routes about other vehicles in the system. Such a system can then be understood and work as a multi-agent system. Nevertheless, presently a centralized control of the transportation systems with fixed tracks and lower autonomy is preferably used in practice. In such a case vehicles are equipped only with sensors for detecting obstacles, embedded process control unit to follow the fixed tracks, sensors for detecting vehicle positions, and a communication system for the AGV having connection with the central control system. Nowadays almost all AGV systems used, e. g., in automobile industries belong to such a class of transportation systems and their design is performed by solving the above mentioned three global aspects separately and heuristically. Naturally then, the solution to reach the goal depends considerably on the specified transportation class and applied optimality criterion as shown by, e. g., Taghaboni-Dutta and Tanchoco [4] and Moorthy et al. [5].

Manuscript received July 12, 2006; revised December 15, 2006. This work was partly supported by the Ministry of Education of the Slovak Rep., under Grant No. 1/3100/06.

Branislav Hrúz is with Inst. of Control and Industrial Informatics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovičova 19, 812 19 Bratislava, Slovak Republic.

Given the transportation structure and tasks or jobs, the basic problem is to find an optimal path in the sense of a criterion for an AGV having to execute an assigned task without collisions with other AGVs. There are two main criteria used in AGVS:

a) minimum task execution time; and

b) minimum traveled task distance.

The first case requires working with time in the control policy. An excellent method for this case is the free window method proposed by Kim and Tanchoco [6] for a class of AGVS. It is based on searching a time optimal path for the new actual transportation task in the free windows labeled mathematical graph representing an actual AGVS transportation situation. The search is realized using the Dijkstra's shortest path algorithm in directed labelled graphs. Paths of the active AGVs are not re-planned so that the global time optimal performance is not considered. The work [6] motivated other researchers further to develop better methods. For example, Maza and Castagna [7] solved another serious problem of the method described in [6] concerning small robustness of the solutions because the routing plan of the system thwarts when the routing plan is not kept precisely. It is an often encountered situation in practice that the planned times are not kept because of the unexpected obstacles in the tracks, variations in the vehicle drive parameters, etc. A new start after the system break-down can be complicated and therefore each system stop can be very expensive. The work [7] improved the Kim and Tanchoco's method using two algorithms to deal with the situation when the AGV system is subject to random contingencies.

A main difference between the basic criteria a) and b) introduced above is in that AGV waiting somewhere in the system does not worsen its behavior according to the second criterion.

In general, transportation systems can be described as discrete event dynamic systems (DEDS). The states are given by positions of the used transportation means and transitions by passes between positions. Consequently, the tools used for modeling and control of DEDS such as Petri nets [8-12], Grafcet [13], reactive flow diagrams [11], state-charts [14-15], are applicable for the transportations systems, as well. Such tools serve firstly for the behavior and function specification of the whole system including its control and secondly for the specification of the control subsystem in a feedback structure "controlled system ⇔ control system". The models are adapted into the form of so-called synchronized models or the models interpreted for control. In such form they enable to generate correct reactive programs for automatic control in an easy way [16]. This is the way we follow in this paper. Solutions presented here are based on the transportation system models using colored Petri nets. Such models serve for the conflict-free process control of the AGV transportation systems and for the optimal AGV routing with respect of the chosen optimality criteria.

2. Specification of Transportation Control Problem

In what follows the transportation system layout with fixed tracks is considered it ws determined in the frame of a manufacturing system. It can be realized as a rail system, inductive wire guidance system or similar [17]. A certain number of AGVs is presumed to operate in the considered transportation system.

The function of AGV systems is to execute various transfers of objects between particular positions in the given manufacturing system. There are two groups of the positions: parking ones and processing ones. The latter are those at loading and unloading devices, at production stations etc. Let the transfers between the positions be denoted as transportation tasks. An AGV system services all the required transportation tasks. Hereafter the transportation function is solved for manufacturing systems. Analogous principles are ascertainable in DEDS of different physical or technological nature.

In the next considerations the tracks are assumed to be divided into zones or sections. Such an arrangement enables to apply a standard rule of the safety, namely that only one vehicle can be in a zone or section. It helps to achieve a primary goal of the AGV system – a collision-free transport operation.

It is assumed that a complete transportation system is built up of two basic elementary modules. The first building module is a switch in the form of single track branching as it is depicted in Fig. 1(a) where the switch is denoted as SWi. The particular forms and dimensions of the switches depend on technological requirements of the considered transportation system. The main parameters of a switch are lengths of the switch tracks, namely distances $dSWn_1n_2$ and $dSWn_1n_3$ where n_1, n_2, and n_3 are the number of the connected plain zones. The second elementary building block is a plain zone without any branching in form of a straight or a curve line as shown in Fig. 1(b).

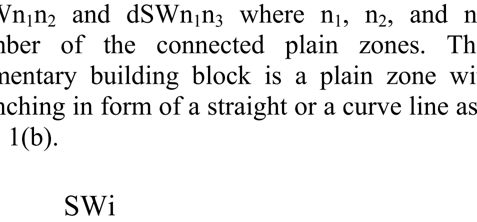


Fig. 1. Elementary modules of the transportation layout.

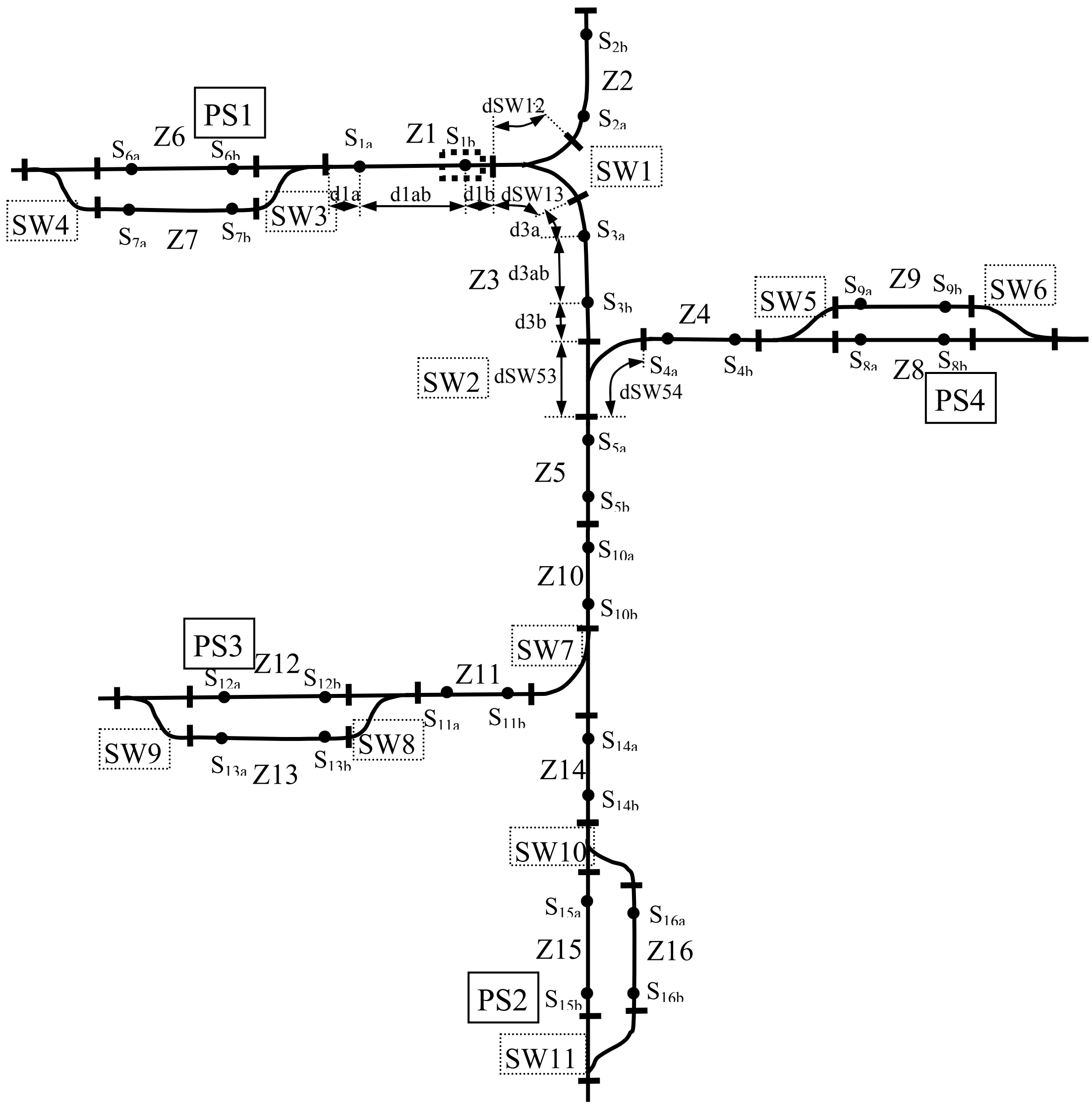


Fig. 2. Transportation layout built up of the elementary modules.

The sensors S_{ja} and S_{jb} represent the checking points for AGV positions. There are two checking points in each zone. The sensors are located in the positions dja and djb according to the necessary locations along the processing stations. Only one processing station or parking position is allowed in one zone. If there is no processing station in a zone the sensors are located in safe distance from the zone ends with respect to the known length of vehicles and their movements in the vicinity of the zone ends. The same holds for the sensor locations associated with the processing stations. Vehicles can stop only in positions at the sensors usually such that a sensor is in the middle of the vehicle. Obviously other arrangements could be easily adopted. The main dimension parameter for a plain zone is its length, e. g., dja, djab, and djb in Fig. 1(b). Each switch in the transportation system must be connected just to one plain zone. On the other hand the same plain zone can be connected at the other side to another switch. A plain zone can be connected to another plain zone. Only one AGV is allowed to be present in a zone. The same holds for each switch in the system. No AGV stopping is permitted in switching areas, i.e. stopping there is an abnormal state.

Fig. 2 shows a part of a transportation layout arrangement built up of the elementary modules. There are eleven switches SW1, SW2,…, SW11 and sixteen plain zones Z1 through Z16. Z3 is an example of the plain zone connected to the different switches. The connection of two plain zones is illustrated by Z5 and Z10.

As mentioned before some points of the zones equipped with sensors are used for the processing stations. It is recommended in this design procedure to plan detours for processing stations in order to prevent obstacles when an AGV is in the zone servicing a processing station. The second recommendation is to reserve special zones for parking free vehicles. To add for such parking zones detours again improves the passage ability of the transportation system.

The AGV motion can be controlled via switching power on/off in the rail tracks or by a communication of the control signals to the vehicles through rails or by some wireless technique.

An important feature of the manufacturing system transportation is the direction of the vehicle movements. There are two possibilities:

1) In all zones or in part of them the bi-directional movements of the vehicles are allowed; and

2) in all zones only unidirectional movements are allowed.

The movement freedom has a decisive influence on the behavior of the system and solution of its control. Usually the first case is intended to obtain a transportation system with more alternative routes and detours while the second one has rather specific connections between the respective zones with few alternative connections. The described diverse transportation arrangements require also different and rather distinctive modeling and algorithmic approach to the control solution.

The function of the transportation is, of course, subordinated to the work-pieces routings according to the required job scheduling in a particular FMS. The following functions can be distinguished:

1) A job schedule and resulting routes are à priori given and they are repeated cyclically in the manufacturing process. The transportation task within the manufacturing jobs is specified by the transfers from a zone to another. The assignments of vehicles to the transfers are fixed; and

2) transportation requests are not given in advance. They occur spontaneously at the beginning and during the manufacturing execution. A vehicle is assigned just for one transfer and after its execution the vehicle is free.

A routing optimization is desirable in both cases. One of the most used optimization criteria is minimizing the transportation time. As discussed earlier another possibility is to minimize the traveling distance and for this criterion waiting of AGV in some position during the transportation job execution does not worsen the transportation result while in the first one it does. In case 1 the solution more or less can be calculated off-line in advance. In case 2 the solution should be calculated during the production run. Transportation durations and/or processing operation durations influence the optimization in both cases. Optimality of the transportation tasks should be understood properly. In case 2, when a new transportation job occurs it is scheduled according to real-time AGV situation. One approach is to solve the problem keeping the schedules of other AGVs. Much more complicated approach would be to search for the solution of the new job with simultaneous re-scheduling of the other AGV in order to achieve a better result. On the other hand the cyclic repetition of a job schedule and its corresponding transportation tasks can be realized with the same methods as in case 2. The transportation tasks are applied in a given order and their scheduling is each time calculated in the real-time situation. Such approach is more robust with respect to various disturbances of the transport in the system as iit is the case when the whole scheduling for all repeated jobs is calculated in advance.

Finally summarize the specifications valid for the method developed in this paper:

- Given fixed track transportation system;
- Transportation system is divided into zones where only one AGV can be at a time;
- Two AGV position sensors are located in each zone;
- AGV bidirectional movement;
- Spontaneous transportation jobs; and
- Transportation time minimization of the actual transportation job to be executed without changing the routes of other AGVs.

3. Process Level Model Based On The Colored Petri Nets

The AGV transportation process running according to the planned routes should be guarded and controlled at the process level correctly without collisions and just in time. Optimal transport routes are planned collision-free, but in reality they can be harmed by disturbances such as suddenly appearing obstacles, deviation in the planned AGV velocities and other. The necessary control functions can be realized at the process level with help of a suitable model. If a deviation in a planned route occur, the process level control system should initiate re-planning of the routes according to the actual AGV situation.

Modeling and control at the process level for the above specified class of AGV transportation systems is based on the colored Petri nets [18]. A modified definition of the colored Petri nets is used as follows.

Definition.

A colored Petri net is the tuple

$$CPN = \left(P, T, \Sigma, F, C, G, LP, LT, E, I\right) \qquad (1)$$

where

P is a finite set of places

T is a finite set of transitions

Σ is a finite and non-empty set of the color sets where a color set is a set of elements given by defined properties

F is a relation $F \subseteq (P \times T) \cup (T \times P)$, $P \cap T = \varnothing$; the elements (p,t) and (t,p) are called arcs for $p \in P, t \in T$

C is a color function $C : P \to \Sigma$

G is a guard function

$$G : T \to \begin{Bmatrix} \text{set of logic expressions containing} \\ \text{Petri net marking and functions} \\ \text{of external variables} \end{Bmatrix}$$

LP is a label function for places, $LP : P \to D_P$, where D_P is a set of values

LT is a label function for transitions $LT : T \to D_T$, where D_T is a set of values

E is a logic expression function $E : F \to SE$ (set of logic expressions)

M_0 is an initial marking

In the definition a marking M is a function that maps the places of CPN into the elements of the color sets corresponding to places according to the function $C(p)$ and to the null element $NULL$ (it is 0 or a tuple of 0) so that $M(p) \in C(p) \cup \{NULL\}$. If $M(p) = c$ then it is said that a token with color c is in p. It is assumed a simplification for the purpose of the work reported in this paper that tokens in each place form a set and not a multiset [16]. Place p is a pre-place of transition t if $(p,t) \in F$ and a post-place if $(t,p) \in F$.

An expression given by the mapping E of an arc represents a condition based on the tokens located in the pre-place of the considered transition. The expression defines an element c of the color set $C(p)$, i.e. $c \in C(p)$. If in a transition's pre-place p there is no token with color c from the color set $C(p)$, then the condition is not fulfilled. If conditions for all arcs are fulfilled and the guard function is true, the considered transition is enabled. Following the general rule for the Petri nets the tokens from pre-places are removed; and according to the transition-post-place arc expression, the resulting tokens are deposited in the post-places. The initial marking states what tokens and where are located at the beginning.

LP in the definition represents a frame for enabling the association of several data values with the places. Similarly it is for the labeling LT.

A basic building module in the form of the colored Petri net for the AGV transportation class under study is depicted in Fig. 4 for a typical elementary part of the transportation system depicted in Fig. 3. The denotation of the elements is the same as in Fig. 2. Again Z1, Z2 and Z3 are zones or sections where only one AGV is allowed to be present. The sensors S_{1a}, S_{1b}, etc. represent the checking points for the AGV positions. The sensors are located in safe positions d1a, d1b, etc. from the zone ends with respect to the known length of vehicles. Vehicles can stop in positions at the sensors when a sensor is in the middle of the vehicle. As mentioned before their stop positions must be safe for the other vehicles moving in the neighborhood. The distances d1ab, d2ab etc. are known, as well. No AGV stopping is permitted in the switching areas.

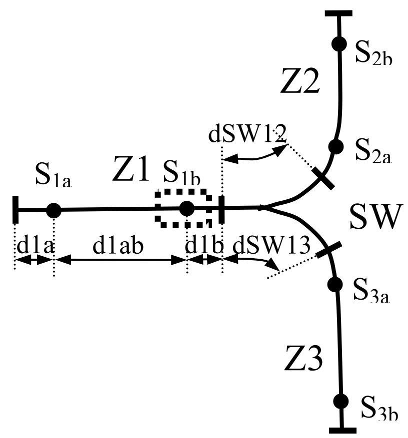
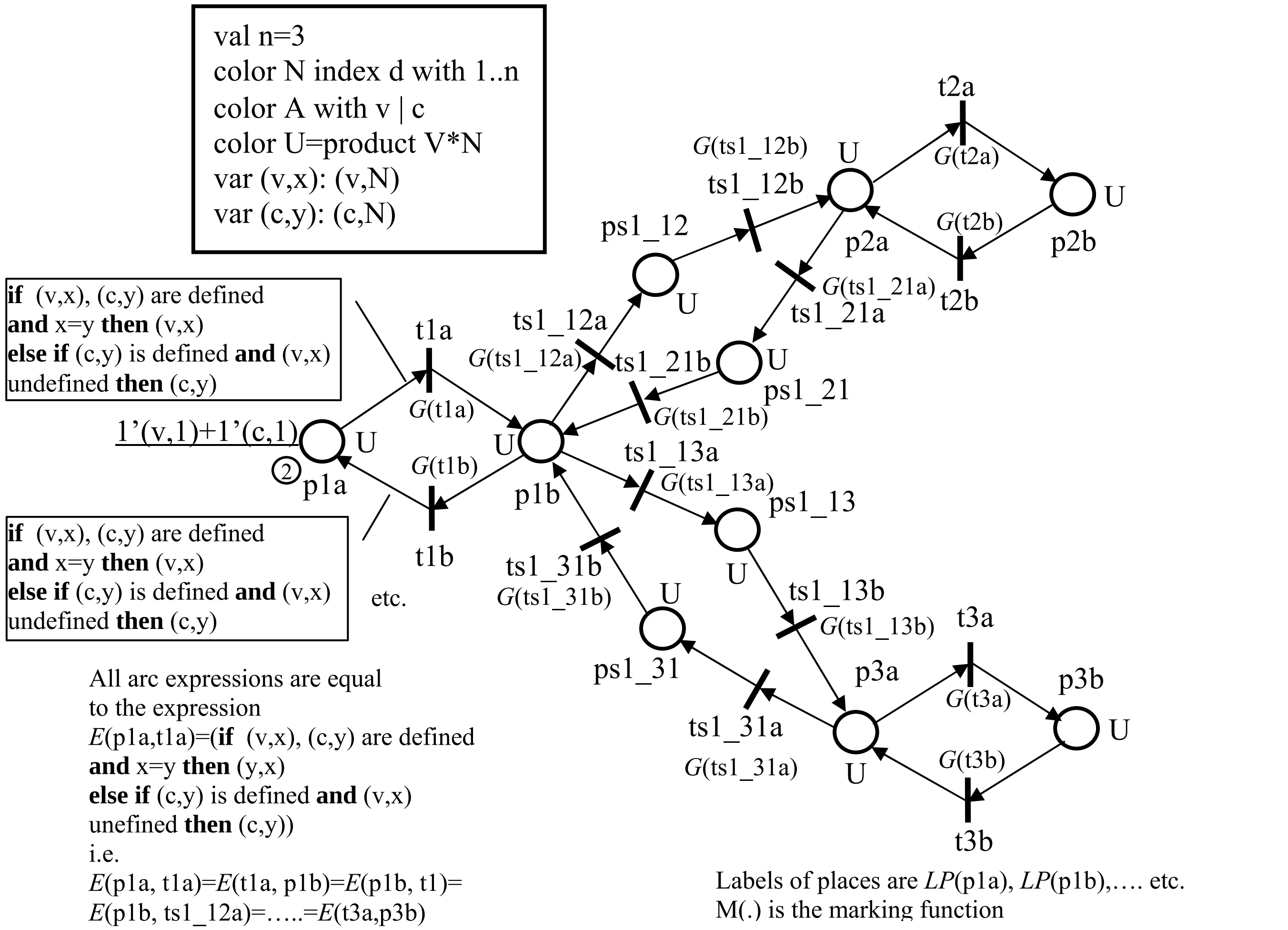


Fig. 3. Elementary module of the transportation layout.

Generally bi-directional movements of the AGV are assumed, which can be easily restricted to uni-directional ones.

Suppose for illustration that three AGV are operating in the system. Variables (v,x), (v,y) are set according to the color tokens that are present in the considered place. Two dimensional variable (v,x) consists of the constant v (color v) and variable $x \in \{1,2,...,n\}$. (v,y) is defined similarly. If no token of the color (v,N) is present in the place, the variable (v,x) is for the considered arc and place undefined. Similarly it is for (c,y).

The colors (v,1), (v,2), (v,3) and (c,1), (c,2), (c,3) correspond to three AGVs, namely No 1, 2, 3, when n=3. M(.) is the marking function. Suppose for example that a token with color (v,1) and a token with color (c,1) are in p1a. The variables x and y in the respective arc expression are such that E(p1t, t1a)=((v,1) and (c,1) are defined and 1=1 then (v,1)). It means that a condition for firing t1a is met because in p1a is a token with color (v,1). Namely, the arc expression is evaluated with respect to the token located in p1a. Another condition for firing t1a is represented by the guard function G(t1a). Suppose further that w1a-1b=T (truth value) and because M(p1a)={(v,1),



G(t1a)=(w1a-1b=T) ∧ {[M(p1a)=(v,N) ∧ M(p1b)=(0,0)] ∨ [M(p1a)=(c,N) ∧ m1b=T]}
G(t1b)=(w1b-1a=T) ∧ {[M(p1b)=(v,N) ∧ M(p1a)=(0,0)] ∨ [M(p1b)=(c,N) ∧ m1a=T]}
G(ts1_12a)=(w1b-s1_12=T) ∧ {[M(p1b)=(v,N)∧ M(ps1_12)=(0,0)] ∨ [M(p1b)=(c,N) ∧ m2a=T]}
G(ts1_12b)=(ws1_12-2a=T) ∧ {[M(ps1_12)=(v,N)∧ M(p2a)=(0,0)] ∨ [M(ps1_12)=(c,N) ∧ m2a=T]}
G(ts1_21a)=(w2a-s1_21=T) ∧ {[M(p2a)=(v,N)∧ M(ps1_21)=(0,0)] ∨ [M(p2a)=(c,N) ∧ m1b=T]}
G(ts1_21b)=(ws1_21-1b=T) ∧ {[M(ps1_21)=(v,N)∧ M(p1b)=(0,0)] ∨ [M(ps1_21)=(c,N) ∧ m1b=T]}
G(ts1_13a)=(w1b-s1_13=T) ∧ {[M(p1b)=(v,N)∧ M(ps1_13)=(0,0)] ∨ [M(p1b)=(c,N) ∧ m3a=T]}
G(ts1_13b)=(ws1_13-3a=T) ∧ {[M(ps1_13)=(v,N)∧ M(p3a)=(0,0)] ∨ [M(ps1_13)=(c,N) ∧ m3a=T]}
G(ts1_31a)=(w3a-s1_31=T) ∧ {[M(p3a)=(v,N)∧ M(ps1_31)=(0,0)] ∨ [M(p3a)=(c,N) ∧ m1b=T]}
G(ts1_31b)=(ws1_31-1b=T) ∧ {[M(ps1_31)=(v,N)∧ M(p1b)=(0,0)] ∨ [M(ps1_31)=(c,N) ∧ m1b=T]}
G(t2a), G(t2b), G(t3a), G(t3b) is created analogously as G(t1a).

T means the true value of logic variables

Fig. 4. Colour Petri net for the basic building module.

(c,1)} and M(p1b)=(0,0) the token with color (v,1) is removed from p1a and deposited in p1b. The arc expression in this particular case obviously does not change the color of the token to be deposited in the post-place. m1a, m1b etc. are logic variables corresponding to signals from the vehicle position sensors. m1a=T if a vehicle is present at the sensor S1a and analogously the logic variable m1b is true when S1b signals the presence of vehicle No 1 at this sensor. An expression *LP*(p1b) associated with p1b is *LP*(p1b)=MOV(v,N). In this particular case *L*(p1b)=MOV(v,1). This is a command for moving vehicle No 1. The command is released when a token with color (v,N) comes in p1b. The direction of the movement is evaluated and the required direction for the vehicle is chosen. When a token with color (c,1) comes in p1b, the situation is analyzed and the vehicle is stopped or its movement continues.

4. AGV Route Planning

The case of the AGV route planning for spontaneous transportation tasks occurring during the system operation is treated in this section. It covers the case of the fixed repeating tasks given in advance, too.

In the route planning process the minimum execution time of the actually planned transportation task or job is considered as an optimality criterion. Then obviously, time aspects of the AGV movements are decisive ones for the AGV route planning. A constant velocity for all AGVs in the system is assumed as a basic approach to the AGV planning problem. It means for instance that the velocity is the same in plain and switch sections. That assumption can be easily replaced with one stating that velocities are constant but different for respective AGVs and even in addition that the velocity is different in plain and switch sections.

It is recommended for a better performance of the transportation system that each section located by a processing station has a parallel detour. The same situation is with the parking sections. Such an arrangement is more effective as that when free AGV are left in any section. Apropos, the sections reserved for parking can be used for maintenance, accumulator charging, cleaning etc.

The transport layout built up according to rules suggested in Section 2 can be represented by an oriented labeled mathematical graph. Let it be called the structure graph. The graph nodes correspond to the sensor locations and form the set

$$ND = \{S_{1a}, S_{1b}, S_{2a}, S_{2a}, ..., S_{na}, S_{nb}\} \quad (1)$$

where

$$ND = N_1 \cup N_2 \cup N_3,$$
$$N_1 \cap N_2 = \varnothing,$$
$$N_1 \cap N_3 = \varnothing,$$
$$N_2 \cap N_3 = \varnothing$$

N_1, N_2, and N_3 are subsets containing the parking AGV locations, processing station locations, and the rest ordinary nodes, respectively. Nodes S_{1a} and S_{1b} are a pair belonging to one zone, similarly S_{2a}, S_{2b} up to S_{na}, S_{nb}. Such pairs are called the zone pairs. The required $\rho -$ th transportation task, $\rho = 1,2,...$, input into the system by an operator can be formally described as

$$T_T(\rho): S_j \rightarrow S_k, S_j, S_k \in ND \;\; S_j, S_k \in N_2 \quad (2)$$

If possible, the planning and control system (PCS) processes immediately a new task. If the system is busy, the task is placed into the FIFO organized queue and processed later.

When the processing starts, a new task given by the expression (2) is transformed into three consecutive tasks. The first one is a free AGV call (assuming any is realizable under given conditions in the current circumstances) from a parking section such that it is performed optimally

$$T_P(\rho): S_i \rightarrow S_j, \quad S_i \in N_1 \quad (3)$$

It is done by evaluating all free AGVs beginning with the closest parking AGV and searching for the optimum.

The second one is

$$T_T(\rho): S_j \rightarrow S_k, S_j, S_k \in N_2 \quad (4)$$

The task $T_T(\rho)$ is to be realized with the same AGV as the $T_P(\rho)$ after the AGV loading. The $T_T(\rho)$ planning starts after the ready signal given by the operator of the processing station.

The third one is

$$T_E(\rho): S_k \rightarrow S_l, S_l \in N_1 \quad (5)$$

Planning of the $T_E(\rho)$ starts after the AGV unloading and releasing a ready signal from the station corresponding to node S_k by the operator. PCS searches the optimal way with respect to time for all above defined tasks to be executed while avoiding the AGV collisions. When a route for some task is found and started the PCS picks up the next task waiting in the queue. The arcs of the structure graph are labeled with the values of times needed for transfers of the AGVs from one sensor location to the next one in the transportation system. Each node has a feedback arc going back to the same node. The arc is labeled with value one time unit. An AGV waiting in the node is represented by the oriented path following repeatedly the feedback arc. For instance, a string (directed labeled path) $\sigma = S_{2a}S_{2a}S_{2a}S_{2a}S_{2a}$ means the stay of the AGV in the node S_{2a} for four time units.

The planning process is organized so that the optimal routes and next worse case routes up to the number R among all nodes of the subset N_2 are calculated in advance off-line and stored in the relevant tables. It improves the speed of the planning process in the real-time circumstances. If after the R steps of testing no route can be found, the task is put off to wait in the queue.

A route is given as a string of nodes to be crossed beginning in the starting node of the task and ending in the goal node of the task. For the tasks given before we have

$$ROUTEP(\rho) = S_i S_{i1} S_{i2} \ldots S_j\,, \quad S_{i1}, S_{i2}, \ldots \in ND \quad (6)$$

$$ROUTET(\rho) = S_j S_{j1} S_{j2} \ldots S_k\,, S_{j1}, S_{j2}, \ldots \in ND \quad (7)$$

$$ROUTEE(\rho) = S_k S_{k1} S_{k2} \ldots S_l\,, S_{k1}, S_{k2}, \ldots \in ND \quad (8)$$

The AGV arrival times corresponding to the routes (6) through (8) are

$$TIMEP(\rho) = \tau_{i1}\tau_{i2}\ldots\tau_j\,, \quad \tau_{i1}, \tau_{i2}, \ldots, \tau_j \in I^+ \quad (9)$$

$$TIMET(\rho) = \tau_{j1}\tau_{j2}\ldots\tau_k\,, \quad \tau_{j1}, \tau_{j2}, \ldots, \tau_k \in I^+ \quad (10)$$

$$TIMEE(\rho) = \tau_{k1}\tau_{k2}\ldots\tau_l\,, \quad \tau_{k1}, \tau_{k2}, \ldots, \tau_l \in I^+ \quad (11)$$

The time values are given as the positive integers meaning the number of the properly chosen time units with respect to dynamical properties of the transportation system. The times $\tau_{i1}, \tau_{j1}, \tau_{k1}$ can respect the AGV acceleration. Purposefully more time points can reflect the acceleration.

For a required task the optimal route is chosen and in the actual transport situation a checking procedure should be activated to check whether the route is realizable. If answer is no the next best route is to be checked etc. up to the R steps, where the number R is rationally chosen. The first realizable route passing through the check is approved and put into execution. If no realizable route is found in the R steps, the task is put into the waiting queue.

If an AGV reaches its goal location and rests there for loading/unloading or for parking, the location is occupied. The set of the current occupied locations is denoted

$$OC = \{S_{o1}, S_{o2}, \ldots, S_{o|OC|}\},\ S_{o1}, S_{o2}, \ldots, S_{o|OC|} \in ND \quad (12)$$

The optimal route search can be formulated as follows. For the set *ND* given by the expression (1) a task

$$T_T(\rho_1) = S_j \to S_k \quad (13)$$

is required. For that task, the route

$$ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k \quad (14)$$

is found as the best in the actual transport situation. To (13) corresponds

$$TIMET(\rho_1) = \tau_{j1}\tau_{j2}\ldots\tau_k \quad (15)$$

The time string (15) is compared and checked on realizeability with all time strings of all other routes or to be more precise with the rest of the routes to be executed from the current time denoted by t_{now}, i. e. with

$$TIMET(x)\ ,\ x \in \{\rho_2, \rho_3, \ldots, \rho_{a0}\} \quad (16)$$

If $ROUTET(\rho_1)$ is not realizeable, it is cancelled and the next best route from S_j to S_k is taken and subjected to the same checking procedure.

Assume further that

$$T_T(\rho_2) = S_w \to S_v \quad (17)$$

$$ROUTET(\rho_2) = S_w S_{w1} S_{w2} \ldots S_v \quad (18)$$

$$TIMET(\rho_2) = \varphi_{w1}\varphi_{w2}\ldots\varphi_v \quad (19)$$

In (19) the time points are denoted by φ to stress the different time values for the different tasks. If (18) is in the run we have rather to do with the part rest route

$$PROUTET(\rho_2) = S_{u1} S_{u2} \ldots S_v \quad (20)$$

and

$$PTIMET(\rho_2) = \varphi_{u1}\varphi_{u2}\ldots\varphi_v \quad (21)$$

which is the string (18) without a string prefix, i.e., without the string part executed before the current time t_{now}. The checking case just before starting (18) can be considered with $u1 = w$ in (20).

The checking algorithm is described on the case specified by the expressions (16) to (21) for $x = \rho_2$.

1. If any node of the oriented path $ROUTET(\rho_1)$ is from the set *OC* then the path is unacceptable.

2. Find common sections or parts (subsequencies) $CS_1,...,CS_s$ of the oriented paths $S_j S_{j1} S_{j2} ... S_k$ and $S_{u1} S_{u2} ... S_v$ so that

$$CS_1 = S_{j,[a1]} S_{j,[a1]+1} ... S_{j,[a2]} = S_{u,[b1]} S_{u,[b1]+1} ... S_{u,[b2]}$$

meaning the same orientation of the paths

or

$$CS_1 = S_{j,[a1]} S_{j,[a1]+1} ... S_{j,[a2]} = S_{u,[b2]} S_{u,[b2]-1} ... S_{u,[b1]}$$

meaning the opposite orientation of the paths,

where indices $(j,[a1])$ and $(j,[a2])$ present the first and last elements, respectively of the first common subsequence for the transport task from location S_j to S_k and similarly for the first common subseqence of the compared route from S_w to S_v, etc., so that

$$CS_2 = S_{j,[a3]} S_{j,[a3]+1} ... S_{j,[a4]} = S_{u,[b3]} S_{u,[b3]+1} ... S_{u,[b4]}$$

or

$$CS_2 = S_{j,[a3]} S_{j,[a3]+1} ... S_{j,[a4]} = S_{u,[b4]} S_{u,[b4]-1} ... S_{u,[b3]}$$

....

$$CS_i = S_{j,[a(2i-1)]} S_{j,[a(2i-1)]+1} ... S_{j,[a(2i)]} = S_{u,[b(2i-1)]} ... S_{u,[b(2i)]}$$

or

$$CS_i = S_{j,[a(2i-1)]} S_{j,[a(2i-1)]+1} ... S_{j,[a(2i)]} = S_{u,[b(2i)]} S_{u,[b(2i)]-1} ... \\ .... S_{u,[b(2i-1)]}$$

....

$$CS_s = S_{j,[a(2s-1)]} S_{j,[a(2s-1)]+1} ... S_{j,[a(2s)]} = S_{u,[b(2s-1)]} ... S_{u,[b(2s)]}$$

or

$$CS_s = S_{j,[a(2s-1)]} S_{j,[a(2s-1)]+1} ... S_{j,[a(2s)]} = \\ S_{u,[b(2s)]} ... S_{u,[b(2s-1)]} \qquad (22)$$

We can see that at least two elements must be in each common subsequence.

3. For all $i = 1,2,...,s$ check the following conditions:

3.1. If CS_i is a common part of the same orientation and $S_{j,[a(2i)]} \neq S_k$ and $S_{u,[b(2i)]} \neq S_v$

and

$$\left(\begin{array}{l} \tau_{j,[a(2i-1)]} \rangle \varphi_{u,[b(2i-1)]+2} \\ \tau_{j,[a(2i-1)]+2} \rangle \varphi_{u,[b(2i-1)]+4} \\ .... \\ \tau_{j,[a(2i)]-1} \rangle \varphi_{u,[b(2i)]+1} \\ \\ \text{or} \\ \\ \tau_{j,[a(2i-1)]+2} \langle \varphi_{u,[b(2i-1)]} \\ \tau_{j,[a(2i-1)]+4} \langle \varphi_{u,[b(2i-1)]+2} \\ .... \\ \tau_{j,[a(2i)]+1} \langle \varphi_{u,[b(2i)]-1} \end{array} \right)$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} ... S_k$ is acceptable, else unacceptable.

3.2. If CS_i is a common part (subsequence) of the same orientation and $S_{j,[a(2i)]} = S_k$ and $S_{u,[b(2i)]} \neq S_v$ and $S_{j,[a(2i-1)]}, S_{j,[a(2i-1)]+1}$ is the zone pair

and

$$\tau_{j,[a(2i-1)]} \rangle \varphi_{u,[b(2i-1)]+2}$$
$$\tau_{j,[a(2i-1)]+2} \rangle \varphi_{u,[b(2i-1)]+4}$$

....

etc. up to $\tau_{j,[a(2i)]}$

and

$$\left(\begin{array}{l} \text{if } S_{j,[a(2i)]} \text{ is an element of a zone pair} \\ \quad \text{and } S_{j,[a(2i)]-1} \text{ is not an element of that pair} \\ \quad \text{and } S_{u,[b(2i)]+1} \neq S_v \\ \quad \text{and } \tau_{j,[a(2i)]} \rangle \varphi_{u,[b(2i)]+2} \\ \text{or} \\ \text{if } S_{j,[a(2i)]} \text{ is an element of a zone pair} \\ \quad \text{and } S_{j,[a(2i)]-1} \text{ is element of that pair, too} \\ \quad \text{and } \tau_{j,[a(2i)]} \rangle \varphi_{u,[b(2i)]+1} \end{array} \right)$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} ... S_k$ is acceptable, else unacceptable.

If CS_i is a common part (subsequence) of the same orientation and $S_{j,[a(2i)]} = S_k$ and $S_{u,[b(2i)]} \neq S_v$

and $S_{j,[a(2i-1)]}, S_{j,[a(2i-1)]+1}$ is **not** the zone pair

and

$\tau_{j,[a(2i-1)]} \rangle \varphi_{u,[b(2i-1)]+1}$

$\tau_{j,[a(2i-1)]+2} \rangle \varphi_{u,[b(2i-1)]+3}$

....

up to $\tau_{j,[a(2i)]}$

and

$$\left(\begin{array}{l} \text{if } S_{j,[a(2i)]} \text{ is an element of a zone pair} \\ \quad \text{and } S_{j,[a(2i)]-1} \text{ is not an element of that pair} \\ \quad \text{and } S_{u,[b(2i)]+1} \neq S_v \\ \quad \text{and } \tau_{j,[a(2i)]} \rangle \varphi_{u,[b(2i)]+2} \\ \text{or} \\ \text{if } S_{j,[a(2i)]} \text{ is an element of a zone pair} \\ \quad \text{and } S_{j,[a(2i)]-1} \text{ is an element of that pair, too} \\ \quad \text{and } \tau_{j,[a(2i)]} \rangle \varphi_{u,[b(2i)]+1} \end{array} \right)$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable.

3.3. If CS_i is a common part (subsequence) of the same orientation and $S_{j,[a(2i)]} \neq S_k$ and $S_{u,[b(2i)]} = S_v$

and $S_{j,[a(2i-1)]}, S_{j,[a(2i-1)]+1}$ is the zone pair

and

$\tau_{j,[a(2i-1)]+2} \langle \varphi_{u,[b(2i-1)]}$

$\tau_{j,[a(2i-1)]+4} \langle \varphi_{u,[b(2i-1)]+2}$

....

etc. up to $\tau_{j,[a(2i)]}$

and

$$\left(\begin{array}{l} \text{if } S_{j,[a(2i)]},\ S_{j,[a(2i)]+1} \text{ is not a zone pair} \\ \quad \text{and } \tau_{j,[a(2i)]+1} \langle \varphi_{u,[b(2i)]} \\ \text{or} \\ \text{if } S_{j,[a(2i)]},\ S_{j,[a(2i)]+1} \text{ is a zone pair} \\ \quad \text{and } S_{j,[a(2i)]+1} \neq S_k \\ \quad \text{and } \tau_{j,[a(2i)]+2} \langle \varphi_{u,[b(2i)]} \end{array} \right)$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable. (Note: if $S_{j,[a(2i)]+1} = S_k$ in the last variant it is unacceptable.)

If CS_i is a common part (subsequence) of the same orientation and $S_{j,[a(2i)]} \neq S_k$ and $S_{u,[b(2i)]} = S_v$

and $S_{j,[a(2i-1)]}, S_{j,[a(2i-1)]+1}$ is **not** the zone pair

and

$\tau_{j,[a(2i-1)]+1} \langle \varphi_{u,[b(2i-1)]}$

$\tau_{j,[a(2i-1)]+3} \langle \varphi_{u,[b(2i-1)]+2}$

....

up to $\tau_{j,[a(2i)]}$

and

$$\left(\begin{array}{l} \text{if } S_{j,[a(2i)]},\ S_{j,[a(2i)]+1} \text{ is not a zone pair} \\ \quad \text{and } \tau_{j,[a(2i)]+1} \langle \varphi_{u,[b(2i)]} \\ \text{or} \\ \text{if } S_{j,[a(2i)]},\ S_{j,[a(2i)]+1} \text{ is a zone pair} \\ \quad \text{and } S_{j,[a(2i)]+1} \neq S_k \\ \quad \text{and } \tau_{j,[a(2i)]+2} \langle \varphi_{u,[b(2i)]} \end{array} \right)$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable.

(Remark: the case $S_{j,[a(2i)]} = S_k$ and $S_{u,[b(2i)]} = S_v$ is excluded; in this case $ROUTET(\rho_1)$ is unacceptable).

4. For all $i = 1,2,\ldots,s$ check the following.

4.1. If CS_i is a common part of the opposite orientation and $S_{j,[a(2i)]} \neq S_k$ and $S_{u,[b(2i)]} \neq S_v$

and

$\tau_{j,[a(2i-1)]} \rangle \varphi_{u,[b(2i)]+1}$

or

$\tau_{j,[a(2i)]+1} \langle \varphi_{u,[b(2i-1)]}$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable.

4.2. If CS_i is a common part of the opposite orientation and $S_{j,[a(2i)]} = S_k$ and $S_{u,[b(2i)]} \neq S_v$

and

$\tau_{j,[a(2i-1)]} \rangle \varphi_{u,[b(2i)]+1}$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable.

4.3. If CS_i is a common part of the opposite orientation and $S_{j,[a(2i)]} \neq S_k$ and $S_{u,[b(2i)]} = S_v$

and

$$\tau_{j,[a(2i)]+1} \langle \varphi_{u,[b(2i-1)]}$$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is acceptable, else unacceptable.

4.4. If CS_i is a common part of the opposite orientation and $S_{j,[a(2i)]} = S_k$ and $S_{u,[b(2i)]} = S_v$

then $ROUTET(\rho_1) = S_j S_{j1} S_{j2} \ldots S_k$ is unacceptable (head-on conflict).

5. Calculation Of The Best Routes

Algorithmically elaborated AGV route planning just introduced is based on the knowledge of the possible routes ordered from the best with respect to its minimum duration for the task to be executed. This problem can be solved in the way described below.

As mentioned earlier a transportation layout can be modeled by a directed labeled graph. The graph labels represent the numbers of the time units required for passing from one system location to another. The labels are obviously positive integers. The incidence matrix corresponding to the graph has the rows and the columns associated with the graph nodes. Let the arc orientation is from the row to the column nodes. Zero value of a matrix entry means that there is no arc from the relevant row node to the column node. The matrix diagonal has value 1 for all entries meaning that each node is provided with a feedback loop labeled with 1. The loops represent AGVs waiting in the location corresponding to the relevant graph node. One transition along the loop means in fact waiting one time unit etc. Repeating of the transition *k* times represents waiting in that location *k* time units. The system control has to ensure stop and waiting in a node by repeating the same node in the realized AGV route.

For the sake of simplicity we omit in the next considerations splitting of nodes into zone pairs (cf. the expression (1)) and then we have the following incidence matrix corresponding to the structure graph

$$G = \begin{array}{c} \\ S_1 \\ \langle S_2 \rangle \\ \cdot \\ \cdot \\ \cdot \\ [S_i] \\ \cdot \\ \cdot \\ \cdot \\ \langle S_n \rangle \end{array} \begin{array}{c} \begin{array}{ccccccccc} S_1 & \langle S_2 \rangle & . & . & . & [S_j] & . & . & \langle S_n \rangle \end{array} \\ \left(\begin{array}{ccccccccc} 1 & w_{12} & . & . & . & w_{1j} & . & . & w_{1n} \\ w_{21} & 1 & . & . & . & w_{2j} & . & . & w_{2n} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ w_{i1} & w_{i2} & . & . & . & w_{ij} & . & . & w_{in} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ w_{n1} & w_{n2} & . & . & . & . & . & . & 1 \end{array} \right) \end{array} \tag{22}$$

where the structure graph nodes are denoted as

$$ND = \{S_1, S_2, \ldots, S_n\} \tag{23}$$

and w_{ijj}; $i = 1,2,\ldots,n$; and $j = 1,2,\ldots,n$, are non-negative integers. Some transportation system locations serve for parking (in (22) denoted by the brackets "$\langle\ \rangle$") and some for processing stations (denoted in (22) by "$[\ ]$"). The transportation tasks are specified and searched for transfers from parking locations to processing ones, between processing locations themselves, and from processing locations to parking ones.

For instance, consider searching the routes for $T_T(\alpha) = S_i \rightarrow S_j$. The algorithm for finding the limited number *NS* of routes ordered according to the transport time is as follows. Symbol $\circ$ denotes a concatenation of strings.

Step 1. Given S_i as starting and S_j as goal nodes, respectively.

Given the positive integer *L* – the maximum number of arcs in the paths and the positive number *NS* – the maximum number of routes.

Specify the sets

$$H_1 = \{T_{11}, T_{12}, \ldots, T_{1(n-1)}\}$$
$$H_2 = \{T_{21}, T_{22}, \ldots, T_{2(n-1)(n-1)}\}$$

....

$$H_L = \{T_{L1}, T_{L2}, \ldots, T_{L(n-1)^L}\}$$

where $T_1, T_2, ..., T_{L(n-1)^L}$ are strings of the graph nodes and set

$T_{11} = S_i, T_{12} = T_{13} = ... = T_{1(n-1)} = T_{21} = ... = T_{L(n-1)^L} = \varepsilon,$

ε is the empty string.

Specify the sets

$D_1 = \{d_{11}, d_{12}, ..., d_{1(n-1)}\}$

$D_2 = \{d_{21}, d_{22}, ..., d_{2(n-1)(n-1)}\}$

....

$D_L = \{d_{L1}, d_{L2}, ..., d_{L(n-1)^L}\}$

where elements of the sets are non-negative integers given as the sums of the path weights corresponding one-by-one to the elements (strings) of the sets $H_1, H_2, ..., H_L$ and set

$d_{11} = d_{12} = ... = d_{1(n-1)} = d_{21} = ... = d_{L(n-1)^l} = 0$

Specify the set of the node strings

$F_{ij} = \{U_1, U_2, ..., U_{NS}\}$

where $U_1 = U_2 = ...U_{NS} = \varepsilon$

and the set of the weight sums corresponding to the strings of the set F_{ij}

$Z_{ij} = \{z_1, z_2, ..., z_{NS}\}$

$z_1 = z_2 = ... = z_{NS} = 0$

Indices i and j in the sets F_{ij} and Z_{ij} stress the correspondence with route $S_i \rightarrow S_j$.

Step 2. Set $s := 0; v := 0; u := 0; A := 1$
Step 3. Set $s := s + 1$
Step 4. If $s == L + 1$
    then goto Step E
  endif
Step 5. Set $r := 0$
Step 6. Set $r := r + 1$
Step 7. If $r == A + 1$
    then $A := u$
      $u := 0$
      order the strings of the sets H_s, D_s according to the increasing values of the corresponding set D_s from minimum
      goto Step 3
  endif
Step 8. Set $LE :=$ the last element of the string T_{sr}
Step 9. If $LE == \varepsilon$
    then goto Step5
  Step 9.1. Set $p := 0$
  Step 9.2. Set $p := p + 1$
  Step 9.3. If $p == n + 1$
      then goto Step 6
      else if $w_{LE,p} \neq 0$
        then if $S_p \neq S_j$
          then $u := u + 1$
            $T_{s+1,u} := T_{sr} \circ S_p$
            $d_{s+1,u} := d_{sr} + w_{LE,p}$
          else $v := v + 1$
            if $v == NS$
              then goto E
              else $U_v := T_{sr} \circ S_p$
                $z_v := d_{sr} + w_{LE,p}$
            endif
          endif
        else goto Step 9.2
      endif
    endif
Step E. Order the elements F_{ij}, Z_{ij} according to the values of the set Z_{ij} beginning from the minimum value.
  END.

The values of L and NS are to be established in a heuristic way according to the dimension and of the transportation system, simulation of the behavior and experience.

If by the execution of a route a collision occurs and process control realizes an exceptional stop of an AGV, re-planning should be done with respect to the actual situation.

6. CONCLUSION

The AGV modeling and control problems in flexible manufacturing systems are introduced in this paper. An applicable approach realizable in practice is described including the time optimal solution that is executable in the real-time circumstances of many manufacturing systems.

REFERENCES

[1] Li, S., Takamori, T. and Tadokoro, S., “Scheduling and re-scheduling of AGVs for flexible and agile manufacturing”, In *Petri Nets in Flexible and Agile Automation,* M.C. Zhou, (Ed.), Kluwer Academic Pulishers, pp.189-205, 1995.

[2] Seo, Y. and P. J. Egbelu, “Integrated manufacturing planning for an AGV-based FMS”, *Int. J. Production Economics,* 60-61, 1999, pp. 473-478.

[3] Ling, Q. and H. Wen-Jing, “Scheduling and routing algorithms for AGVs: a survey,” *Technical Report::CAIS-TR-99-26*, 1999, Centre for Advanced Information Systems, School of Applied Science, Nanyang Technological University.

[4] Taghaboni-Dutta, F. and J. M. A. Tanchoco, “Comparison of dynamic routing techniques for automated guided vehicle system,” *Int.J. Prod. Res.*, Vol. 33, No. 10, Oct. 1995, pp. 2653-2669.

[5] Moorthy, R. L., Hock-Guan, W., Wing-Cheong, Ng., and Chung-Piaw, T., “Cyclic deadlock prediction and avoidance for zone-

controlled AGV system," *Int. J. Production Economics*, Vol. 83, 2003, pp. 309-324.
[6] Kim, C. W. and J. M. A. Tanchoco, "Conflict-free shortest-time bidirectional AGV routing," *Int. J. Prod. Res.*, Vol. 29, No. 12, 1991, pp. 2377-2391.
[7] Maza, S. and P. Castagna, "Robust conflict-free routing of bi-directional automated guided vehicles," *Proc. of the IEEE Conf . on System, Man and Cybernetics*, Tunisia, Hammamet, Oct. 5-9, 2002.
[8] Zhou, M. C. (Ed.), *Petri Nets in Flexible and Agile Automation*, Kluwer Academic Pulishers, Nowell, MA,1995.
[9] Murata, T., "Petri nets: properties, analysis and applications," *Proc. IEEE*, Vol. 77, No. 4, April 1989, pp. 541-580.
[10] Abel, D., *Petri-Netze fűr Ingenieure*, Springer-Verlag, 1990, Berlin.
[11] Hrúz, B. and M. C. Zhou, *Modeling and Control of Discrete Event Dynamic Systems with Petri nets and other tools*, Springer, London, to appear in 2007.
[12] Zhou, M. C. and K. Venkatesh, *Modeling, Simulation, and Control of Flexible Manufacturing Systems*, World Scientific, Singapore, 1999.
[13] David, R. and H. Alla, *Petri nets and Grafcet*, Prentice Hall, New York, 1992.
[14] Harel, D., "Statecharts: A visual formalism for complex systems," *Science of Computer Programming*, North Holland, Vol. 8, 1987, pp.231-274.
[15] Fogel, J., "A statecharts approach to the modeling of discrete manufacturing systems", *Proc. of the 7th Symposium on Computer Aided Syst. Design*, Gent, April 28-30, 1997.
[16] Hrúz, B., Niemi, A., and Virtanen, T., "Composition of conflict-free Petri net models for control of flexible manufacturing systems, *Reprint of the 13th IFAC World Congress*, San Francisco, USA, 1996, Vol. B, pp. 37-42.
[17] Koff, G. A., "Automatic guided vehicle systems: applications, controls, and planning," *Material Flow*, No 4, 1987, pp. 3-16.
[18] Jensen, K., *Coloured Petri Nets*, Springer-Verlag, Berlin. 1997.

Branislav Hrúz received his M. S. degree from Czech Technical University in Prague, Dept. of Electrical Engineering, in 1960, with study specialized on control engineering. He received Ph. D. degree from the Slovak Academy of Sciences, Institute of Technical Cybernetics in Bratislava, in 1968. He joined the Research Institute of Mathematical Machines in Prague in 1960 and then the Institute of Technical Cybernetics of the Slovak Academy of Sciences, Bratislava, Slovak republic in 1963, and the State Forest Research Institute in Bratislava, Control Engineering Laboratory, in 1971, where he lead the first projects of computer automation of the woodworking industry. Since 1991 he is an Associate Professor at the Slovak University of Technology in Bratislava, member of Faculty of Electrical Engineering and Information Technology, Institute of Control and Industrial Informatics. His main interests are in modeling and control of discrete event dynamic systems, threshold logic functions, finite automata, Petri nets and other tools enabling to program the event system control applied in manufacturing, AGV systems, industrial supervisory control and in other fields. Dr. Hrúz is author or co-author of 100+ publications including 3 books and he presented over fifty papers at national or international conferences. He acted as a visiting Professor at the Vienna University in Austria and at the Helsinki University of Technology in Finland.