A NEW HIERARCHICAL CONTROL SCHEME FOR A CLASS OF CYCLICALLY REPEATED DISCRETE-EVENT SYSTEMS

Danjing Li §, Eckart Mayer §, Jörg Raisch §, ‡
§*Systems and Control Theory Group, Max-Planck Institute, Dynamics of Complex Technical Systems*
Sandtorstrasse 1, D-39106 Magdeburg, Germany

‡*Lehrstuhl für Systemtheorie technischer Prozesse, Otto-von-Guericke Universität Magdeburg*
Postfach 4120, D-39016 Magdeburg, Germany

Keywords: Cyclic systems, discrete-event systems, max-plus algebra, min-plus algebra, rail traffic.

Abstract: We extend the hierarchical control method in (Li et al., 2004) to a more generic setting which involves cyclically repeated processes. A hierarchical architecture is presented to facilitate control synthesis. Specifically, a conservative max-plus model for cyclically repeated processes is introduced on the upper level which provides an optimal online plan list. An enhanced min-plus algebra based scheme on the lower level not only handles unexpected events but, more importantly, addresses cooperation issues between sub-plants and different cycles. A rail traffic example is given to demonstrate the effectiveness of the proposed approach.

1 INTRODUCTION

In (Li et al., 2004), a hierarchical two-level control architecture has been introduced for a class of discrete-event systems (DES) without cyclically repeated features (i.e. for systems "running only one cycle"). The upper level of this architecture produces the time optimal plan based on the event time relation represented by a max-plus algebra model. The plan describes the time optimal sequence of events. If an unexpected event happens at any given time, max-plus algebra models are simulated online on the upper level to update the optimal plan and the detailed time specification for every event. Using this information, the lower control level acts as an implementation block.

In practice, many DES applications exhibit cyclically repeated features. In this case, cooperation (and competition) issues between different cycles have to be addressed, which is beyond the scope of the algorithm in (Li et al., 2004). To cope with this situation, the control strategy in (Li et al., 2004) is extended. The resulting hierarchical control scheme is depicted in Fig.1. It is composed of a two-level structure with an independent C/D (continuous/discrete) module. In general, the supervisory block on the upper level will have the goal of determining the sequence of events which optimises the given objective function in the cyclically repeated case. This sequence of events is referred to as the optimal plan. The C/D module is an interface block which transforms information from the (continuous) plant to the timed DES framework employed on the upper level of the proposed control architecture.

A case study representing a simple rail traffic system is used to demonstrate how control is realised in the proposed framework. For this example, events correspond to specific trains crossing specific locations within the track system. A plan specifies a sequence of trains and track segments where trains pass each other. The lower control level generates velocity reference signals for the trains to implement the plan determined by the supervisory block. The C/D block transforms position information into event time information. For other applications, the terms "train" and "track (segment)" have of course to be replaced by different ones. For example, in flexible manufacturing systems, "product" and "machine" can be used instead. In a general context, "train" stands for "user", "track (segment)" for "resource".

This contribution is organised as follows. Section 2 summarises how to generate the set of feasible plans for a DES without cyclically repeated processes and extends this method to the more general setting with cyclically repeated features. This section also explains how the optimal plan is then chosen in an online procedure. The C/D block is described in Section 3. Section 4 explains how the plan generated at the upper level can be implemented on the lower level with the help of min-plus algebra. A simple case study is given in Section 5 to illustrate the effectiveness of the proposed approach.

Li D., Mayer E. and Raisch J. (2005).
A NEW HIERARCHICAL CONTROL SCHEME FOR A CLASS OF CYCLICALLY REPEATED DISCRETE-EVENT SYSTEMS.
In *Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics - Signal Processing, Systems Modeling and Control*, pages 30-36
DOI: 10.5220/0001182800300036

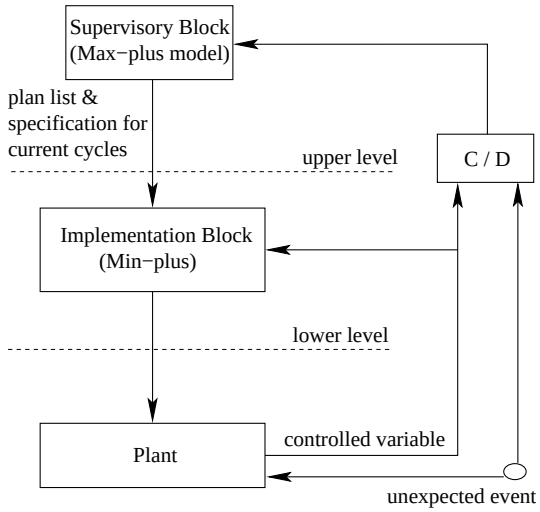

Figure 1: Control structure

2 SUPERVISORY LEVEL

Max-plus algebra e.g. (Baccelli et al., 1992) is a structure defined on the set $\mathbb{R}^* = \mathbb{R} \cup \{-\infty\}$, with $\max$ and $+$ as the two binary operations $\oplus$ and $\otimes$, respectively. $\epsilon := -\infty$ is the additive identity in the max-plus algebra. $e := 0$ is the multiplicative identity.

Based on the max-plus algebra model of the system, the supervisory level generates the set of all feasible plans (i.e. all feasible sequences of events) as an offline task. During runtime, the supervisory level determines the time optimal plan based on real time information provided by the C/D module. The difference between systems with and without cyclicly repeated features lies in their different max-plus models. We first summarise the simpler case.

2.1 Max-plus Model for non-cyclic Systems

Non-cyclic systems can be represented by the implicit max-plus algebra model (Baccelli et al., 1992):

$$\underline{X} = A_0 \otimes \underline{X} \oplus B \otimes u, \quad (1)$$
$$Y = C \otimes \underline{X}. \quad (2)$$

with

$$A_0^* = I \otimes A_0 \otimes A_0^2 \otimes \cdots \otimes A_0^n \otimes A_0^{n+1} \otimes \cdots, \quad (3)$$

(1), (2) can be converted to an explicit max-plus model

$$\underline{X} = A_0^* \otimes B \otimes u, \quad (4)$$
$$Y = C \otimes A_0^* \otimes B \otimes u, \quad (5)$$

Here, the i-th component of $\underline{X}$, $\underline{x}_i$, represents the earliest possible time for event i to occur. u and Y contain time information of "start" and "finish" events, and can be interpreted as system input and output, respectively.

The elements of matrix A_0 represent the minimum time distances that have to pass between events, e.g. $(A_0)_{21}$ implies that event 2 may not happen earlier than $(A_0)_{21}$ time units after event 1 happened. In this contribution, $(A_0)_{ij} \in \mathbb{R}^+ \cup \{-\infty, 0\}$. Matrix A_0 is the max-plus sum of A_{01} and A_{02}, i.e.

$$A_0 = A_{01} \oplus A_{02}, \quad (6)$$

where the former represents a property of the system and is a fixed matrix, while the latter is related to the shared resources and differs for different plans. More specifically, the elements of A_{01} represent the time relation between events determined by physical properties of the system. For example, for rail systems, with a maximum speed assumption in max-plus algebra, the entries of A_0 represent the minimum time needed by trains to pass distances in the network. Thus the elements of A_{01} correspond to lengths of tracks. On the other hand, for any instance of time, a shared track segment can only be occupied by one train. The different occupation orders for trains on shared track segments generate different plans. Different A_{02} matrices correspond to those different plans.

In a directed graph, we call the arcs corresponding to A_{01} elements "travelling arcs" while the arcs corresponding to A_{02} elements are called "control arcs". For a given train track network, the travelling arcs and therefore the entries of matrix A_{01} are fixed. Fig. 2 shows the directed graph (including all possible control arcs) of a simple network with two single-line track segments and two trains, where train 1 moves from right to left and train 2 moves in opposite direction. For single-line segment I, the control arc labelled a_1, with $a_1 \geq 0$, represents the fact that the earliest time instant at which train 2 may occupy this segment is a_1 time units after train 1 has emptied it. The control arc labelled b_1 states the reverse sequence. Accordingly, only one of those two arcs can exist in any one plan. Instead of erasing a non-existent arc from a graph, we can also label it with $\epsilon = -\infty$. Hence, by definition, non-existing arcs have weight ϵ. Similarly, in segment II, the arcs labelled a_2 and b_2 represent two possibilities. Detailed information about how to find all feasible A_{02} (i.e. all feasible plans) can be found in (Li et al., 2004). In particular, it was shown there that

$$\exists k \leq n, A_0^k(i,i) > \epsilon \Leftrightarrow A_{02} \text{ is infeasible.} \quad (7)$$

2.2 Max-plus Model for Cyclicly Repeated Systems

For system with cyclicly repeated behaviour, several cycles may exist concurrently. Therefore, control

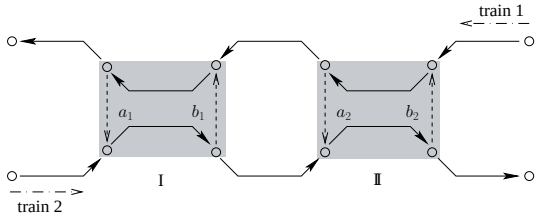

Figure 2: A network with all possible control arcs

conditions have to be extended such that trains from different cycles will not occupy the same track segment simultaneously. Hence, additional control arcs have to be introduced. Also, additional travelling arcs are needed to represent the passage of trains into subsequent cycles. An arc that connects events related to the same cycle is a "zero order arc", an arc that connects events related to two sequential cycles is a "first order arc". In general, there can also be "second order arcs" and so on. To keep our example reasonably simple, we introduce the following conservative condition:

For each track segment shared by several cycles, trains in a latter cycle may not occupy it unless all trains in the previous cycle have left it.

With the above assumption, there is no need to introduce arcs of higher order than 1. For the simple rail traffic network in Fig. 3, which was shown in (Li et al., 2004), "first order control arcs" ensure the safe resource sharing between cycles and "first order travelling arcs" ensure that each train starts its new journey (cycle $k+1$, $k \geq 1$) a specified number of time units after its arrival at its destination in cycle k. As a review of the example, the network involves 3 trains and 3 tracks. Initially, train 1, 2 and 3 are located at the end points of the 3 tracks, i.e. A, B and C, respectively. The trains move along the tracks in the directions shown in Fig. 3. In the middle of both track AO and track CO, double-line track segments are available, which make it possible that two trains pass each other between points N_1 and M_1 or N_2 and M_2, respectively. In (Li et al., 2004), all trains stopped after they arrived at their destinations (i.e. after one cycle). In this contribution, each train will start a new cycle, e.g. a given number of time units after train 1 arrived at C, it will start the next route to go to B, which represents a new cycle. Fig. 4 shows the different kinds of control and travelling arcs for this network. Note that although there are optional control arcs of zero order, all first order control arcs are fixed according to the assumption.

Thus, for systems with cyclicly repeated behaviour, in addition to the zero order matrix A_0, we now have a first order matrix A_1. For cycle k, the implicit max-

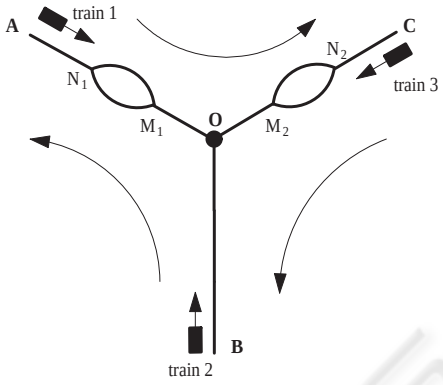

Figure 3: A simple rail traffic network

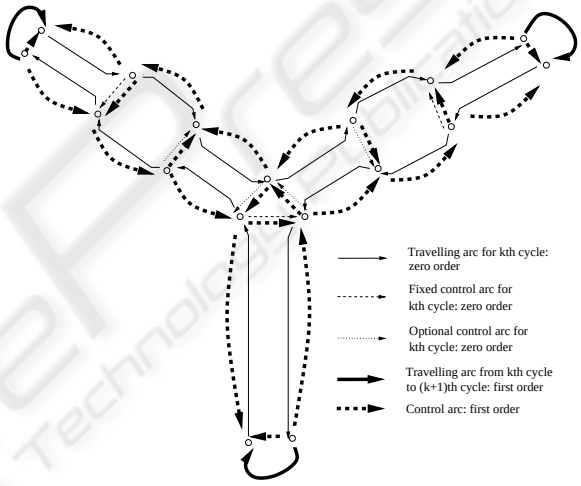

Figure 4: Control arcs and travelling arcs.

plus algebra model is:

$$\underline{X}(k) = A_0(k)\underline{X}(k) \oplus A_1\underline{X}(k-1) \oplus Bu \quad (8)$$
$$Y(k) = C \otimes \underline{X}(k) \quad (9)$$

As for A_0, the matrix A_1 is the max-plus sum of two matrices A_{11} and A_{12}, i.e.

$$A_1 = A_{11} \oplus A_{12}, \quad (10)$$

where the elements of A_{11} correspond to "first order travelling arcs" representing the time needed for a train to start a new cycle after finishing the previous cycle. The elements of A_{12} correspond to the control arcs for trains belonging to two sequential cycles respectively. Unlike A_{02}, A_{12} cannot represent different choices; therefore, A_1 is a constant matrix while A_0 will depend on the specific plan to be implemented in the k-th cycle and is therefore varying with k.

Repeated insertion of (8) into itself provides:

$$\begin{aligned}\underline{X}(k) &= A_0(k)\otimes\underline{X}(k)\oplus A_1\otimes\underline{X}(k-1)\oplus B\otimes u\\ &= A_0(k)\otimes[A_0(k)\otimes\underline{X}(k)\oplus A_1\otimes\underline{X}(k-1)\\ &\quad\oplus B\otimes u]\oplus A_1\otimes\underline{X}(k-1)\oplus B\otimes u\\ &= A_0^2(k)\otimes\underline{X}(k)\oplus[A_0(k)\oplus I]\otimes A_1\\ &\quad\otimes\underline{X}(k-1)\oplus[A_0(k)\oplus I]\otimes B\otimes u\\ &= A_0^2(k)\underline{X}(k)\oplus[A_0(k)\oplus I]A_1\underline{X}(k-1)\\ &\quad\oplus[A_0(k)\oplus I]Bu\\ &\vdots\\ &= A_0^n(k)\underline{X}(k)\oplus[A_0^{n-1}(k)\oplus\cdots\oplus A_0(k)\\ &\quad\oplus I]A_1\underline{X}(k-1)\oplus[A_0^{n-1}(k)\oplus\cdots\\ &\quad\oplus A_0(k)\oplus I]Bu \qquad (11)\end{aligned}$$

As for physically meaningful systems, $A_0^n(k)$ will only contain ϵ elements, the last identity represents an explicit max-plus algebra model:

$$\begin{aligned}\underline{X}(k) &= A_0^*(k)\otimes A_1\otimes\underline{X}(k-1)\oplus\\ &\quad A_0^*(k)\otimes B\otimes u \qquad (12)\\ Y(k) &= C\otimes\underline{X}(k), \qquad (13)\end{aligned}$$

where

$$A_0^*(k)=[A_0^{n-1}(k)\oplus\cdots\oplus A_0(k)\oplus I]. \qquad (14)$$

Assume that the system input u contains initial values of lower bounds for the state vector $X(k)$, i.e. $B=I$ ($B_{ii}=e, B_{ij}=\epsilon$ for $i\neq j$), $u=\underline{X}\_in(k)$ (see Section 3 for further information), (12) and (13) become

$$\begin{aligned}\underline{X}(k) &= A(k)\otimes\underline{X}(k-1)\oplus\\ &\quad A_0^*(k)\otimes\underline{X}\_in(k) \qquad (15)\\ Y(k) &= C\otimes\underline{X}(k) \qquad (16)\end{aligned}$$

where $A(k)$ depends on $A_0(k)$:

$$A(k)=A_0^*(k)A_1. \qquad (17)$$

For a given system with cyclicly repeated behaviour, (7) can be applied to eliminate infeasible plans and corresponding max-plus models.

2.3 Optimal Plan List

After all feasible plans have been generated, the supervisory level may simulate feasible max-plus-models (15), (16) over the required total number of cycles to determine the optimal sequence of $A(k)$, i.e. the optimal plan list. This is initially done in an offline fashion, i.e. before the system is started. The objective function is typically evaluated from the output vector. For example, for rail track systems, the (offline) objective function could be

$$J_{offline}=\min_{\substack{\text{all feasible}\\ \text{plan lists}}}\Big(\bigoplus_i Y_i(N)\Big) \qquad (18)$$

where N is the required total number of cycles and $Y_i(N)$ is the arrival time of train i in the N-th cycle. As $\oplus$ represents max, the physical meaning is to choose the plan list giving the minimum final arrival time, i.e. the minimal arrival time of the last train in the last cycle. Since it is offline simulation, $\underline{X}\_in(1)$ only carries the starting event time. Furthermore, $(\underline{X})_i(0)$ is set to $\epsilon=-\infty$. The resulting optimal plan list is then implemented via the lower control level.

Max-plus simulation in this step is obviously based on the assumption that there is no unexpected event and that each train either waits for a synchronisation condition to be met or otherwise moves at its maximum velocity. Thus if any unexpected incident happens, it may affect the travelling time of trains either directly by blocking them or indirectly via synchronisation with other trains. If this happens, the runtime event time will not match the event time calculated by a-priori model simulation any more. To address such difficulties, an online simulation is integrated to update the aforementioned optimal plan list.

At time t_k, for each of the concurrently existing cycles l (e.g. $l=k,k+1$), $[\underline{X}(l)](t_k),[Y(l)](t_k)$ will be updated by using (15), (16) with $[\underline{X}\_in(l)](t_k)$ provided by the C/D block.

The online objective may be the same as the offline objective (18), i.e.

$$J_{online1}=J_{offline}. \qquad (19)$$

It may also be different from the offline objective, for example, it may state that after unexpected delays, all trains have to catch up with the timetable corresponding to the original offline plan list as fast as possible. In this case,

$$J_{online2}=\min_{\substack{\text{all feasible}\\ \text{plan lists}}}K_\triangle \qquad (20)$$

where $K_\triangle$ is the index of the earliest cycle in which all delayed trains can start (or end) their cycle trips according to the timetable, i.e.

$$\begin{cases}\forall i,\triangle_i(l)=0. & l\geq K_\triangle\\ \exists i,\triangle_i(l)>0. & l<K_\triangle\end{cases} \qquad (21)$$

$$\triangle_i(l)=t_{start_i}(l)-timetable_i(l) \qquad (22)$$

where $t_{start_i}(l)$ is the actual cycle starting time of train i in cycle l, $timetable_i(l)$ is the cycle starting time (according to the timetable) of train i in cycle l.

In brief, the supervisory level involves both offline and online parts. The offline process is used to find all feasible plans based on the system topology and an offline optimal plan list, which can be interpreted as a timetable. The online part updates the feasible plan set (see (Li et al., 2004)), the optimal plan list and provides the time specification $\underline{X}(k)$ to the lower level.

3 C/D BLOCK

In order to reschedule the plan (using (15)-(16)) in a real time fashion, the state vector needs to be reinitialised according to the current status (at time t_k) of the trains for online model simulation. In other words, the continuous variables will be transformed into a reinitialised state vector. The reinitialised state vector $\underline{X}\_in(t_k)$ for the current cycle is:

$$\underline{x}\_in_j(t_k) = \begin{cases} x_j & \text{if } j \text{ is a past event} \\ t_k + t_{rest} & \text{if } j \text{ is a next event, for unblocked trains} \\ t_{rel} + t_{rest} & \text{if } j \text{ is a next event, for blocked trains} \\ \epsilon & \text{otherwise} \end{cases} \tag{23}$$

where x_j is the actual time at which event j happened. t_{rest} is the minimum time still needed before the next event j may happen. It introduces continuous variable information. For a rail track system, if a train has to go the distance $d_j(t_k)$ before it reaches the location associated with the next event j and if the maximum train velocity is v_{max},

$$t_{rest} = \frac{d_j(t_k)}{v_{max}}. \tag{24}$$

t_{rel} is the estimated release time for the train currently blocked. If this time cannot be estimated beforehand, plan rescheduling will be based on the assumption of immediate release, i.e.

$$t_{rel} = t_k. \tag{25}$$

The supervisory level together with the C/D block serves a model predictive control purpose, and a receding horizon concept is involved. This relates our work to (De Schutter et al., 2002; De Schutter and van den Boom, 2001; Van den Boom and De Schutter, 2004), where model predictive control is combined with max-plus discrete-event systems.

4 IMPLEMENTATION LEVEL

The output of the supervisory level is the time specification $\underline{X}(k)$ for the events of the currently existing cycles (e.g. $j = k, k+1$). $\underline{X}(k)$ provides the earliest possible time for each event due to the maximum speed assumption under max-plus. According to EPET (Earliest Possible Event Time) specifications, trains will always move at maximum speed, or otherwise stop to wait until the synchronisation conditions are met. This is undesirable from an energy saving point of view. An overall optimisation approach, minimising total energy consumption for all trains, yields a large nonlinear problem, which, at the moment, seems unrealistic to solve. Instead, a suboptimal approach is used here to determine a velocity signal separately for each train. This is done as follows:

First, a Latest Necessary Event Time (LNET) for each train is derived as an upper bound for the event times. For systems with non-cyclicly repeated features, this is done in such a way that the event time of the final event for each train according to EPET will be met, in addition, it is also ensured that LNET does not violate the EPET schedule of the other trains. For systems with cyclicly repeated features, LNET should not violate the EPET schedule of the sequential cycle. Min-plus algebra, the dual system of max-plus algebra (Baccelli et al., 1992; Cuninghame-Green, 1979; Cuninghame-Green, 1991), is adopted to produce the exact LNET specification.

Suppose $Q(j)$ is the event index set for all final events of trains in cycle j and $P_m(j)$ is the index set for all events related to train m in cycle j, then at time t_k the LNET specifications $[\overline{X}_m(j)](t_k)$ for train m in cycle j can be calculated as

$$\begin{aligned}[\overline{X}_m(j)](t_k) \;=\; & (-(A_0(j)^*)^T) \otimes' [\underline{XR}_m(j)](t_k) \\ & \oplus' \; (-A^T(j+1)) \otimes' \underline{X}(j+1)\end{aligned} \tag{26}$$

where

$$[(\underline{XR}_m)_i(j)](t_k) = \begin{cases} [\underline{x}_i(j)](t_k), & i \in Q(j) \text{ or } i \notin P_m(j) \\ +\infty, & \text{otherwise.} \end{cases} \tag{27}$$

Thus, for each train m, its EPET $\underline{X}(j)$ and LNET $\overline{X}_m(j)$ are generated. Within this corridor, the velocity signal can be optimised locally for each train. Under ideal conditions, an energy optimal trajectory results from minimising

$$J_m = \int v_m^2 dt. \tag{28}$$

For the problem of driving a train from one station to the next, an energy optimal driving style consists of four parts: maximum acceleration, speedholding, coasting and maximum breaking (Franke et al., 2002; Howlett et al., 1994, e.g.). On a long journey the speedholding phase becomes the dominant phase. Then, assuming the journey must be completed within a given time, the holding speed (i.e. the energy optimal velocity) is approximately the total distance divided by the total time (Howlett, 2000). In the following, we neglect acceleration and braking effects and suppose the velocity between two sequential events is the straight-line velocity (i.e. speedholding mode). Then, (28) becomes:

$$J_m = \sum_{i=1}^{n} \frac{(S_i - S_{i-1})^2}{(t_i - t_{i-1})} \tag{29}$$

s.t.

$$t_0 = 0 \tag{30}$$

$$t_i = t_{i-1} + \frac{S_i - S_{i-1}}{v_m} \tag{31}$$

$$t_{1E} \leq t_1 \leq t_{1L} \tag{32}$$

$$t_{2E} \leq t_2 \leq t_{2L} \tag{33}$$

$$\vdots$$

$$t_{(n-1)E} \leq t_{n-1} \leq t_{(n-1)L} \tag{34}$$

$$t_{nE} = t_n = t_{nL} \tag{35}$$

where $S_0 = 0$, S_i $(i = 1, 2, \cdots, n)$ is the distance from the current position of train m to the place where event i happens. t_{nE} and t_{nL} are EPET and LNET of event n, respectively. The numerical solution gives the energy optimal trajectory.

5 RAIL TRAFFIC CASE STUDY

The effectiveness of the hierarchical control architecture proposed in the previous sections is illustrated by a small rail traffic example which is depicted in Fig. 3. There are 3 feasible plans in this case study (see previous work in (Li et al., 2004)).

In a first step, a fixed optimal-time plan list is derived from offline simulations. The system is expected to run based on this list. If unexpected events happen and block some train, the supervisory level will decide whether and how to change the plan list.

Suppose the total number of cycles is $N = 5$. Fig. 5 shows the movements of the trains in a normal situation. The optimal plan list is [1 2 2 1 2]. In case of a disturbance (here train 3 is blocked on track M_2N_2 during the time interval $[6, 46]$), the supervisory level checks if it is necessary to modify the plan list using $J_{online1}$. This scenario is presented in Fig. 6 and Fig. 8, respectively. In Fig. 6, the blocking time is known beforehand. With this blocking time information, the supervisory level changes the plan list immediately to reduce the delay time caused by the obstacle. The required 5 cycles finish in about 219 time units. As a comparison, Fig. 7 keeps the original plan list and the required 5 cycles finish in about 228 time units. If the blocking time is unknown beforehand, the supervisory level also keeps the original plan list until the status of the trains (include the blocked train) evolve to such an extent that another plan list optimises the online objective $J_{online1}$. In this example, as shown in Fig. 8, with the new plan list, [1 2 1 2 2], the required 5 cycles still can finish in 219 time units.

Under the same blocking situation, with different online objectives, the supervisory level may change to different plan list. If, for example, the required number of cycles is 7, and train 3 is blocked during the time interval $[6, 26]$, and the online cost function

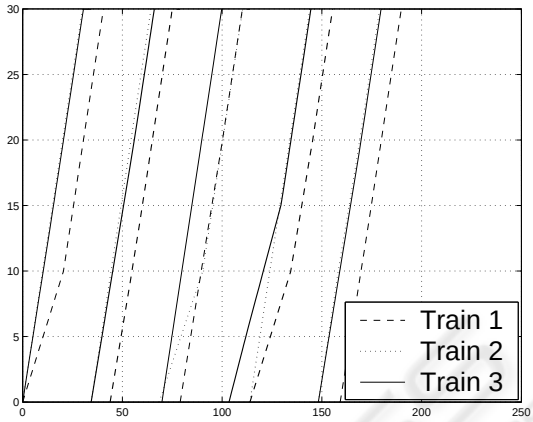

Figure 5: Simulation result under normal situation, plan list: [1 2 2 1 2]

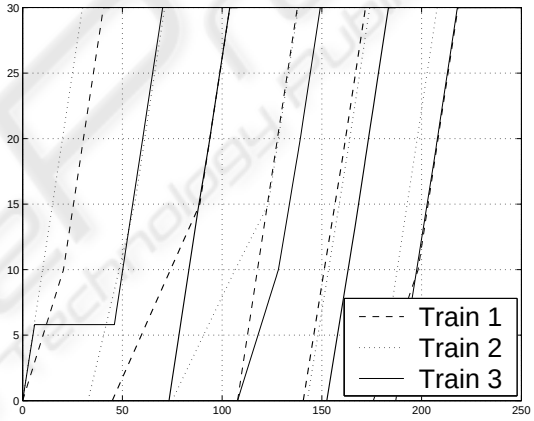

Figure 6: Blocking time is known beforehand, new plan list: [3 2 1 2 2]

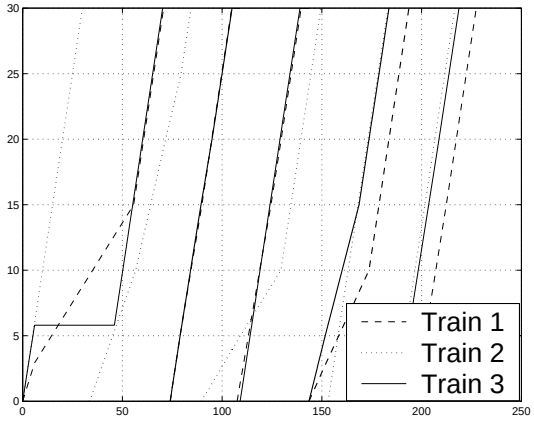

Figure 7: Blocking time is known beforehand, stick to original plan list: [1 2 2 1 2]

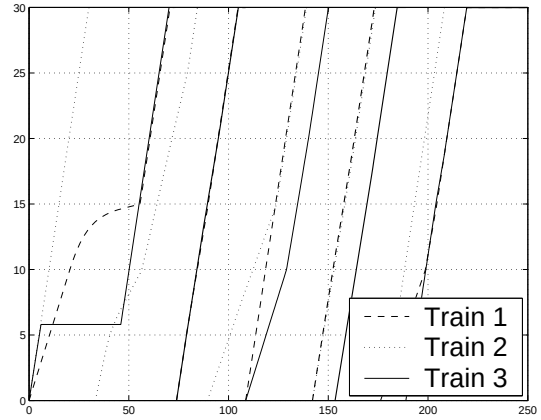

Figure 8: Blocking time is unknown beforehand, new plan list: [1 2 1 2 2]

$J_{online2}$ is used, the supervisory level changes the plan list from [1 2 2 1 2 2 2] to [1 2 1 2 2 2 2], and the offline timetable is recovered after the $6th$ cycle. With $J_{online1}$, the plan list is changed to [1 2 1 2 2 1 2], which minimises the required time. However, for this policy, the timetable is not recovered.

6 CONCLUSION

This contribution proposes a hierarchical control architecture for a class of discrete-event systems with cyclicly repeated features and illustrates it through a small rail traffic example. Based on a max-plus algebra model, an upper level supervisory block ensures the optimal sequence of train movements and the optimal sequence of cycle plans even under disruptive conditions. A lower level implementation block provides reference velocity signals for each train. By exploiting the remaining degrees of freedom, it reduces overall energy consumption. The implementation policy is generated by use of the dual min-plus algebra model. Simulation results for the rail traffic example show the effectiveness of the approach. We expect the method to be also useful in other DES applications which exhibit cyclicly repeated features, such as flexible manufacturing systems and chemical batch processing plants.

REFERENCES

Baccelli, F., G.Cohen, G.J.Olsder, and J.P.Quadrat (1992). *Synchronization and Linearity*. John Wiley and Sons, New-York.

Cuninghame-Green, R. (1979). *Minimax algebra*. Springer-Verlag, Berlin. Vol. 166 of Lecture Notes in Economics and Mathematical Systems.

Cuninghame-Green, R. (1991). Minimax algebra and applications. *Fuzzy Sets and Systems*, 41:251–267.

De Schutter, B. and van den Boom, T. (2001). Model predictive control for max-min-plus-scaling systems. In *Proceedings of the 2001 American Control Conference*, pages 319–324, Arlington, Virginia.

De Schutter, B., van den Boom, T., and Hegyi, A. (2002). A model predictive control approach for recovery from delays in railway systems. *Transportation Research Record*, no.1793:15–20.

Franke, R., Meyer, M., and Terwiesch, P. (2002). Optimal control of the driving of trains. *At - Automatisierungstechnik*, 50(12):606–613.

Howlett, P. (2000). The optimal control of a train. *Annals of Operations Research*, 98:65–87.

Howlett, P., Milroy, I., and Pudney, P. (1994). Energy-efficient train control. *Control Eng. Practice*, 2(2):193–200.

Li, D., Mayer, E., and Raisch, J. (2004). A novel hierarchical control architecture for a class of discrete-event systems. In *7^{th} IFAC Workshop on DISCRETE EVENT SYSTEMS. Reims, France*, pages 415–420.

Van den Boom, T. and De Schutter, B. (2004). Modelling and control of discrete event systems using switching max-plus-linear systems. In *7^{th} IFAC Workshop on DISCRETE EVENT SYSTEMS. Reims, France*, pages 115–120.