Delft University of Technology

Delft Center for Systems and Control

Technical report 16-018

Model predictive control of an automated storage/retrieval system*

D.J. Nativ, A. Cataldo, R. Scattolini, and B. De Schutter

If you want to cite this report, please use the following reference instead:
D.J. Nativ, A. Cataldo, R. Scattolini, and B. De Schutter, "Model predictive control of an automated storage/retrieval system," *Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management and Control (MIM 2016)*, Troyes, France, pp. 1335–1340, June 2016.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.51.19 (secretary)
fax: +31-15-278.66.79
URL: `http://www.dcsc.tudelft.nl`

*This report can also be downloaded via `http://pub.deschutter.info/abs/16_018.html`

Model Predictive Control of an Automated Storage/Retrieval System

Danielle J. Nativ [*] **Andrea Cataldo** [**] **Riccardo Scattolini** [**]
Bart De Schutter [*]

[*] *Delft Center for Systems and Control, Delft University of Technology, the Netherlands*
[**] *Department of Electronics, Information and Bioengineering, Politecnico di Milano, Italy*

Abstract: Discrete-event systems occur often in the manufacturing field, but due to the characteristics of these systems Model Predictive Control (MPC) is not frequently used for them. This paper approaches MPC of an Automated Storage/Retrieval System (AS/RS) by using Mixed Logical Dynamical (MLD) modelling. Propositional calculus is used to transform the non-linear dynamic model equations and constraints of the AS/RS into the MLD form. We consider a performance index that makes sure that customer demand is satisfied as soon and as efficiently as possible. The MLD model and performance index are reformulated as an integer linear programming problem. A case study is performed on a laboratory stacker crane, and the simulations results illustrate the good performance of the control algorithm.

Keywords: Control applications, Manufacturing systems, Production processes, Warehouse automation, Self-organising storage, Model-based control, Predictive control, Discrete-event systems, Integer programming

1. INTRODUCTION

Over the years most industries have become more and more demanding due to the complexity of production processes, and tighter rules and regulations. These refined processes demand accurate control. Model Predictive Control (MPC) (Camacho and Bordons, 1997) is a control method widely used in the process industry nowadays since it has proven itself capable of dealing with complex systems (Qin and Badgwell, 2003). This is due to the ability of MPC to enable reformulation of control problems into optimisation problems, which gives the opportunity to explicitly add constraints on the control inputs and the controlled variables.
Nevertheless, model predictive controllers are mostly used with continuously varying systems and are less frequently used for discrete-event systems. This is why in the manufacturing field, which is often characterised by discrete-event systems, MPC is barely used. This is quite surprising considering that in the manufacturing field high performance and efficiency are required (Flegel, 2014). Some applications of MPC in the manufacturing field have been investigated by Vargas-Villamil and Rivera (2000), focusing on the application of MPC in semiconductor manufacturing lines.
The scarce use of MPC for discrete-event systems can be explained by the fact that these systems are characterised by integer or Boolean decision variables (Xi et al., 2013). For discrete-event systems large combinatorial optimisation problems often need to be solved on-line because of these variables, which is seen as a computational bottleneck.

One specific type of systems used in the advanced manufacturing field are automated storage/retrieval systems (AS/RSs) (Lee, 1997). The introduction of AS/RSs improved inventory management and control, increased storage capacity and reliability, and reduced unnecessary labour costs. One of the main components of an AS/RS is the storage/retrieval crane, which is used to pick up and drop off items. Research shows that there are many ways to address issues with the control of AS/RSs, e.g. Roodbergen and Vis (2009) mention several methods for storage assignment, and Bessenouci et al. (2012) focus on the estimation of the travel time of the crane.
The complexity of control of AS/RSs is related to the number of depots and the storage policy of the system. Many papers use a random storage policy, which allows a pallet to be stored randomly on any available storage location. Due to the flexibility of such a policy, the number of possible solutions enlarges, which causes a higher complexity to optimally solve the storage assignment problem. Many researchers solve this problem with heuristics (Han et al., 1987; Mahajan et al., 1998; Dooly and Lee, 2008; Ávila et al., 2015). A solution to problems with a dedicated storage policy is described by Gharehgozli et al. (2014), where in contrary to a random storage policy, a dedicated storage policy predefines a unique storage location for each storage request. Gharehgozli et al. (2014) show that this problem can be solved in polynomial time by using the fact that the crane always returns to a depot.

In this paper a different approach is suggested to control AS/RSs with a random storage policy. Since MPC has many advantages it is desirable to use it for the control of AS/RSs. This paper overcomes the complex application

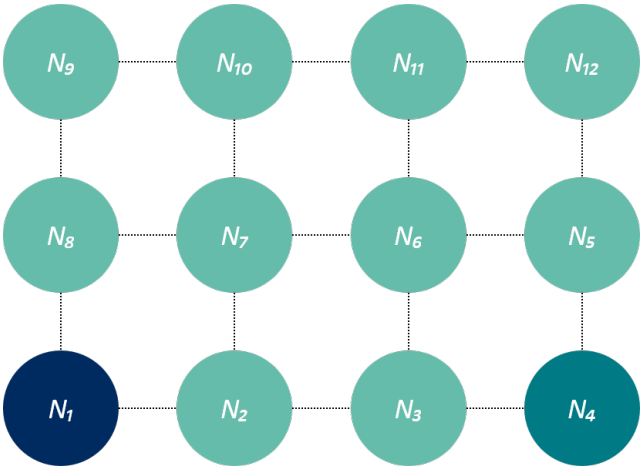

Fig. 1. Schematic representation of the storage facility

of MPC to AS/RSs by describing the system as a Mixed Logical Dynamical (MLD) system. The transformation to an MLD system gives the advantage that the control problem can be expressed as a mixed integer programming problem. The combination MLD-MPC has been applied before (Du et al., 2009; Beccuti et al., 2005; Groot et al., 2013), but to our best knowledge, only few applications for discrete-event manufacturing systems have been reported (Cataldo and Scattolini, 2014; Cataldo et al., 2015), and never for AS/RSs.

This paper is organised as follows. In Section 2, a description of the stacker crane is given. Next, Section 3 describes the dynamical model and its constraints, followed by the MLD-MPC problem in Section 4. Then, Section 5 discusses the results of a case study with a laboratory stacker crane, resulting in conclusions and recommendations in Section 6.

2. AUTOMATED STORAGE/RETRIEVAL SYSTEM

An AS/RS is used for automated storage of materials. In general the system consists of two main components: the storage facility and the crane.
The storage facility can store pallets at different locations, which can be represented by a set of nodes N, see Figure 1 for an example. One of these nodes can be defined as the source node, $N_\mathrm{s} \in N$, where new pallets arrive for storage. Another node can be defined as final storage node, $N_\mathrm{f} \in N$, where the customers can pick up their order. When this node is filled, temporary storage places can be used to store the extra delivered pallets. The temporary storage places are represented by all nodes that are neither the source node nor the final storage node, resulting in the subset $N_\mathrm{T} \subset N$ with temporary storage nodes.
For this paper a single-level-deep stationary storage facility is assumed, which means that the crane can directly reach the stored pallets. Solutions to multiple-levels-deep storage problems are discussed by Yu and De Koster (2012). This paper will focus on single unit-load aisle-captive AS/RSs. This means that there is one crane per storage facility that cannot leave its designated aisle, and that cannot carry more than one pallet at a time (Roodbergen and Vis, 2009). The crane can move through three Cartesian axes to pick up the pallets and to put them down in storage where needed. The x- and y-axes are used to move the crane to the right position, the z-axis is used by a piston to load or unload a pallet. Each event step the crane can only travel to an adjacent node. Therefore, for every node N_i a set $S_i \subset N$ of adjacent nodes is defined, e.g. $S_3 = \{N_2, N_4, N_6\}$ according the set-up depicted in Figure 1.
For an AS/RS different types of pallets can be defined as $p \in P$. We define P as follows: $P = \{2, \cdots, |P|+1\}$, where $|P|$ is the number of different pallet types handled by the system, since the values of 0 and 1 will be used in the model to define the absence of the crane in front of a node and the absence of a pallet on the crane, respectively. It is assumed that every pallet has the same final storage node $N_\mathrm{f} \in N$.

3. DYNAMIC MODEL

In this section the dynamic model of the AS/RS will be described. First the state and decision variables are defined, and then the dynamic equations and constraints are derived.

State variables

For each node $N_i \in N$, with $i \in \{1, \ldots, n\}$ (i.e. $n = |N|$) two state variables can be defined to describe the system. The first variable is $P_i(k)$, which represents the state and the position of the crane (i.e. does the crane carry a pallet or not, and is the crane in front of node N_i or not). The second state variable, $G_i(k)$ represents the state of the nodes (i.e. does node N_i store a pallet or not). These variables are defined in the following way:

$$P_i(k) = \begin{cases} p & \text{, if the crane is at } N_i \text{ with a pallet type } p \\ 1 & \text{, if the crane is at } N_i \text{ without a pallet} \\ 0 & \text{, otherwise} \end{cases}$$

$$G_i(k) = \begin{cases} p & \text{, if } N_i \text{ stores a pallet of type } p \\ 0 & \text{, otherwise} \end{cases}$$

Decision variables

The following decision variables can be defined to represent, together with the state variables, the dynamical model of the AS/RS:

$$u_{ij}(k) = \begin{cases} 1 & \text{, the crane moves from } N_i \text{ to } N_j \\ 0 & \text{, otherwise} \end{cases}$$

$$v_i(k) = \begin{cases} 1 & \text{, a pallet is loaded from the crane onto } N_i \\ 0 & \text{, otherwise} \end{cases}$$

$$w_i(k) = \begin{cases} 1 & \text{, a pallet is loaded from } N_i \text{ onto the crane} \\ 0 & \text{, otherwise} \end{cases}$$

Dynamic equations

Now, assuming one movement per event step, the following explicit discrete state space model can be defined, which represents the states of the system at event step $k+1$:

$$P_i(k+1) = P_i(k) - \sum_{j \in S_i} u_{ij}(k) P_i(k) + \sum_{j \in S_i} u_{ji}(k) P_j(k) - v_i(k)[P_i(k) - 1] + w_i(k)[G_i(k) - 1] \quad (1)$$

$$G_i(k+1) = G_i(k) + v_i(k) P_i(k) - w_i(k) G_i(k) \quad (2)$$

Equation (1) shows that there are four different events that can change the state $P_i(k)$ of the system to a different state in the next time step, $P_i(k+1)$:

- The crane moves from N_i to N_j, thus $P_i(k) \geq 1$, $P_j(k) = 0$, and $u_{ij(k)} = 1$.
- The crane moves from N_j to N_i, thus $P_i(k) = 0$, $P_j(k) \geq 1$, and $u_{ji}(k) = 1$.
- The piston loads a pallet from the crane onto N_i, thus $P_i(k) \geq 2$, and $v_i(k) = 1$.
- The piston loads a pallet from N_i onto the crane, thus $P_i(k) = 1$, and $w_i(k) = 1$.

The correctness of (1) can be illustrated as follows. If at event step k the crane stays where it is and a pallet is loaded from the crane onto node N_i, (1) will reduce to $P_i(k+1) = P_i(k) - v_i(k)[P_i(k) - 1]$ since all the other terms of the dynamic equation are equal to zero. As the crane is in front of N_i and holds a pallet of type p, $P_i(k) = p$. Moving the pallet onto the node requires $v_i(k) = 1$, which, according to the dynamic equation, results in the next state $P_i(k+1) = 1$. This means that at event step $k+1$ the crane is still in front of node N_i but does not hold a pallet, which is in line with the definition of $P_i(k)$ given at the start of Section 3. Equation (2) is influenced in a similar way. If the crane moves and the piston does not, the state of the nodes do not change: $G_i(k+1) = G_i(k)$. However, if the piston moves the state of the node changes.

Deliveries and demands

For the source node, and final storage node dynamic equation (2) applies, but some extra rules need to be regarded. The arrivals of new pallets at the source node are predefined e.g. with a uniform distribution. Per event step maximal one pallet arrives. When node N_s is empty, i.e. $G_1(k) = 0$, a new pallet arrives at N_s and $G_1(k+1) = p$, depending on the type of pallet. When $G_1(k) \neq 0$, dynamic equation (2) applies to determine $G_1(k+1)$.
The customer demand, D, which is chosen uniformly distributed over P, influences the final storage node, N_f. When the demand is satisfied the state of final storage node N_f at even step k is $G_\mathrm{f}(k) = D(k)$, next, the pallet will be removed from the final storage node by the customer, and thus $G_\mathrm{f}(k+1) = 0$. When the customer demand is not satisfied G_f will be calculated according to dynamic equation (2).

Constraints

The following constraints, describe the restrictions of the system, and apply to every $N_i \in N$:

- Only one event per time step k may occur:
$$\sum_{j \in S_i} u_{ij}(k) + \sum_{j \in S_i} u_{ji}(k) + v_i(k) + w_i(k) \leq 1 \qquad (3)$$
- If the crane is not at node N_i, the crane cannot move from a node N_i to node N_j:
$$P_i(k) = 0 \rightarrow \sum_{j \in S_i} u_{ij}(k) = 0 \qquad (4)$$
- A pallet cannot be loaded from the crane onto node N_i if the crane is not positioned in front of N_i, carries no pallet, or the node carries already a pallet:
$$(P_i(k) \leq 1) \vee (G_i(k) \geq 1) \rightarrow v_i(k) = 0 \qquad (5)$$
where $\vee$ denotes the disjunction.
- A pallet cannot be loaded from node N_i onto the crane, if the crane is not front of N_i, already carries a pallet, or N_i carries no pallet:
$$(P_i(k) \neq 1) \vee (G_i(k) = 0) \rightarrow w_i(k) = 0 \qquad (6)$$

Note that (4), (5), and (6) result in non-linear constraints. However, these constraints can be reformulated in a linear way by the use of propositional calculus (Raman and Grossmann, 1992; Cavalier et al., 1999). For this extra auxiliary variables are created, as will be illustrated next.

Example. By adding Boolean auxiliary variables to the model, $\delta_i(k)$, (4) is reformulated, resulting in the following expressions:
$$P_i(k) = 0 \rightarrow \delta_i(k) = 1$$
$$\delta_i(k) = 1 \rightarrow \sum_{j \in S_i} u_{ij}(k) = 0$$
For the ease of reading from now on, in this example, $\sum_{j \in S_i} u_{ij}(k) = U_i(k)$.
The following two proposition rules are used to translate the constraints (Bemporad and Morari, 1999).

Proposition 1 Assuming that $x \in X$, where X is a given bounded set, $m = \min_{x \in X} f(x)$, and ϵ a very small positive number, leads to the following statement:
$[f(x) \leq 0] \rightarrow [\delta = 1]$ if and only if $f(x) \geq \epsilon + (m - \epsilon)\delta$.
Let a literal X_i represent a statement which is either true or false, e.g. $x \geq 1$. One can associate to a literal X_i a Boolean (auxiliary) variable $\delta_i \in \{0, 1\}$. If X_i is true, $\delta_i = 1$, and $\delta_i = 0$ otherwise.

Proposition 2 The following expressions and linear constraint can be seen to be equivalent:
$X_1 \rightarrow X_2$ is equivalent to $\delta_1 - \delta_2 \leq 0$.
Since $\min_k P_i(k) = 0$, $P_i(k) = 0 \rightarrow \delta_i(k) = 1$ is equivalent to $P_i(k) \leq 0 \rightarrow \delta_i(k) = 1$. Note that the latter formulation is now in the right form to use with the propositional rule of Proposition 1. This results in the first set of constraints that need to be added to the model:
$$P_i(k) \geq \epsilon - \epsilon\delta_i(k)$$
Note that $m = 0$.
Next, the rule of Proposition 2 is used to add $\delta_i(k) = 1 \rightarrow U_i(k) = 0$ to the model. This results in the following constraint:
$$\delta_i(k) + U_i(k) \leq 1$$
It is left to the interested reader to derive the other constraints with the use of propositional calculus.

4. MLD-MPC

The dynamic model of the AS/RSs, described in the previous section, can be translated into a Mixed Logical Dynamic (MLD) formulation using propositional calculus as explained above. The MLD formulation has the following form:
$$x(k+1) = Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) \qquad (7)$$
$$y(k) = Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) \qquad (8)$$
$$E_1x(k) + E_2u(k) + E_3\delta(k) + E_4z(k) \leq g_5 \qquad (9)$$

where $x(k)$ is the vector of state variables, $u(k)$ the vector of control actions, with elements $u_{ij}(k)$, $\delta(k)$ is the vector of Boolean auxiliary variables, and $z(k)$ is a vector of continuous auxiliary variables.

Objective function

The performance index J is defined at event step k as:

$$J(k) = \alpha_1 J_{\text{node}}(k) + \alpha_2 J_{\text{crane}}(k) + \alpha_3 J_{\text{penalty}}(k) + \alpha_4 J_{\text{pallet}}(k) \tag{10}$$

where $\alpha_1, \alpha_2, \alpha_3$, and α_4 are weight factors. The objective function consists of the following parts:

- J_{node} can be used to give certain nodes priority over others to load or unload them faster, e.g. it is desired to keep the source node empty for new pallets that arrive.
- J_{crane} is used to reduce the total time the crane is carrying a pallet. This is to avoid the system storing a pallet on the crane instead of loading it into the node.
- J_{penalty} penalises unnecessary crane movements.
- J_{pallet} tries to satisfy the customer demand as soon as possible.

The performance index (10) must be minimised under the constraints (3)-(6). Once the sequence of optimal control actions has been computed over a prediction horizon N_{p}, according to the receding horizon approach, only the values of the first event step are applied. The overall procedure is repeated at the next event step.
Objective J_{node} can be used to define preferences for certain nodes. To do so, the following Boolean auxiliary variable is used:

$$\gamma_i(k) = \begin{cases} 1 & \text{, if a pallet is present in node } N_i \\ 0 & \text{, otherwise} \end{cases}$$

By multiplying this variable by a weight factor, the system will try to store pallets in nodes with a lower weight factor rather than a higher weight factor, when minimising the performance index. This formulation is represented by:

$$J_{\text{node}}(k) = \sum_{t=1}^{N_{\text{p}}} \sum_{i=1}^{n} q_i \gamma_i(k+t-1),$$

where q_i is the weight factor corresponding to node N_i. As described before, J_{crane} is used to unload a pallet from a node instead of keeping it stored on the crane. By defining $\eta(k)$, a Boolean auxiliary variable, considering the presence of a pallet on the crane:

$$\eta(k) = \begin{cases} 1 & \text{, if the crane holds a pallet} \\ 0 & \text{, otherwise} \end{cases}$$

this results in

$$J_{\text{crane}}(k) = \sum_{t=1}^{N_{\text{p}}} \eta(k+t-1)$$

To save energy it is desired to minimise the total amount of crane movements. This is represented by J_{penalty}. Defining a penalty of q_{c} and q_{p} for movements of the crane and piston respectively, results in:

$$J_{\text{penalty}}(k) = \sum_{t=1}^{N_{\text{p}}} \Big[q_{\text{c}} \sum_{(i,j) \in N \times N} u_{ij}(k+t-1) + q_{\text{p}} \sum_{i=1}^{n} \big(v_i(k+t-1) + w_i(k+t-1) \big) \Big]$$

The definition of the customer satisfaction in J_{pallet} will be elaborated next. It is desired to satisfy the customer demand as soon as possible. If the pallet type present in final storage node N_{f} is equal to the requested pallet type by the customer, $D(k)$, we set the Boolean auxiliary variable $\varsigma(k) = 1$:

$$\varsigma(k) = \begin{cases} 1 & \text{, if } D(k) = G_{\text{f}}(k) \\ 0 & \text{, otherwise} \end{cases}$$

This can be transformed into a linear form using propositional calculus. The following objective function for the customer satisfaction is proposed:

$$J_{\text{pallet}}(k) = -\sum_{t=1}^{N_{\text{p}}} \varsigma(k+t-1)$$

It is desired to satisfy the customer as soon as possible, which means that we want to maximise $\sum_{t=1}^{N_{\text{p}}} \varsigma(k+t-1)$. Since the total objective function $J_{\text{pallet}}(k)$ will be minimised, the objective function is described by the negative sum of $\varsigma(k)$.
The objective function and MLD model of the system have now been transformed into a integer linear programming problem, which can be optimised using adequate solvers.

5. CASE STUDY

A case study is performed on a laboratory stacker crane to test the MLD-MPC application on an AS/RS. This section first describes the system used and then discusses the simulation results.

Laboratory stacker crane

The crane is located in the laboratory of the Electronics, Automation, and Bioengineering department of the Politecnico di Milano, Italy. It is a scale model of a real stacker crane.
The storage facility can store pallets on three different levels, where each level has four places available for storage. This means that the whole storage facility can take up to twelve pallets (i.e. $n = 12$), see Figure 1. Here N_1 is the source node where the pallets, that need to be stored, are delivered. Node N_4 is defined as the final storage node, where customers can come to pick up their order. There are three different types of pallets used, so $P = \{2, 3, 4\}$.
The temporary storage places are represented by all nodes that are not the source node nor the final storage node. This results in: $N_{\text{s}} = N_1$ is the source node, $N_{\text{f}} = N_4$ is the final storage node, and $N_{\text{T}} = \{N_2, N_3, N_5, N_6, N_7, N_8, N_9, N_{10}, N_{11}, N_{12}\}$ is the set with temporary storage nodes.

Weight factors

Next we will describe what values for the weight factors in the objective function are used. It is desired to remove

a pallet from the source node, N_1 as soon as possible, and therefore q_1 is set higher than the other q_i weight factors of the temporary storage nodes. Next to that, it is preferable to keep the final destination node, N_4, empty as long as an order cannot be satisfied. Therefore it is decided to give q_1 and q_4 a value of 1, and to use a value of 0 for $q_i, i \in N_\mathrm{T}$, since there is no preference in loading or unloading certain temporary storage.
Since the crane movements consume significantly more energy than the piston movements, the weight factor of the crane movement should be set higher than the other, and therefore $q_\mathrm{c} = 2$ and $q_\mathrm{p} = 1$ are chosen.
The weight factors $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are set according to importance of the different objectives. Since customer satisfaction had priority number one, α_4 has a higher value than the other factors. Second most important is satisfaction of J_node, and on a shared third place come J_crane and J_penalty. The chosen values for the weight factors are $\alpha_1 = 1$, $\alpha_2 = 0.12$, $\alpha_3 = 0.1$, and $\alpha_4 = 10$. The prediction horizon N_p has been set to 10.

Simulation results

To be able to better understand the simulation results, an example result is given next.

Example. A basic example of the evolution of the system over time is given in Figure 2. For the ease of understanding, here only 4 nodes of the whole system are depicted, including source node N_1 and final storage node N_4. At event step $k = 1$ the customer at node N_4 demands a pallet of type 4, i.e. $D(1) = 4$. At this time the variable $\varsigma(1) = 0$ since node N_4 is empty, and thus the performance index can be optimised by making $\varsigma(k) = 1$ with k as small as possible. At the same moment, i.e. $k = 1$, there is a pallet of the demanded type available at the source node, N_1, so the optimised control input sequence allows managing the pallet movement step by step over time to final storage node N_4, so that the customer can pick up his order. Note that the crane first moves one node per time step, and then the piston moves to place the pallet in the storage facility (i.e. $v_4(4) = 1$).
Different from the example of Figure 1, if a pallet type p requested from the customer is already stored in a temporary storage node, the MLD-MPC algorithm will control its movement towards the final node N_4. Moreover, in case a pallet p type is requested by the customer and two pallets of the same type are stored in a temporary storage node and in the source node N_1 respectively, then the MLD-MPC algorithm will pick up the pallet from node N_1 since the weight $q_1 = 1$ and the weight related to temporary storage node N_i, $q_i = 0$. By picking up the pallet from the source node the final performance index value will be optimised.
Finally, if no pallet is available to satisfy the customer demand, neither in the source node nor in any temporary storage node, then the MLD-MPC algorithm will not implement any pallet movement even if the performance index value increases over time due to the unsatisfied customer demand at final node N_4. In such a case, as soon as a pallet of the requested type p is placed into the source node, the control will manage its movement as previously discussed.

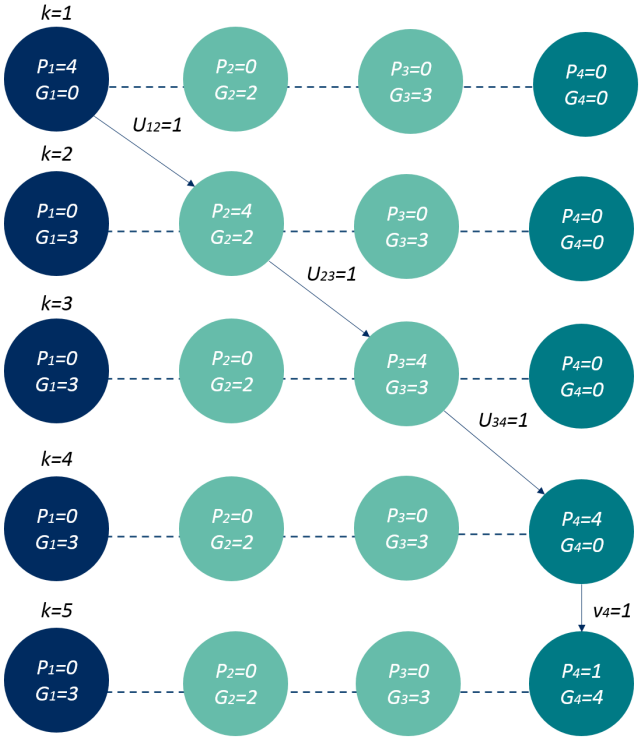

Fig. 2. Example of state transitions

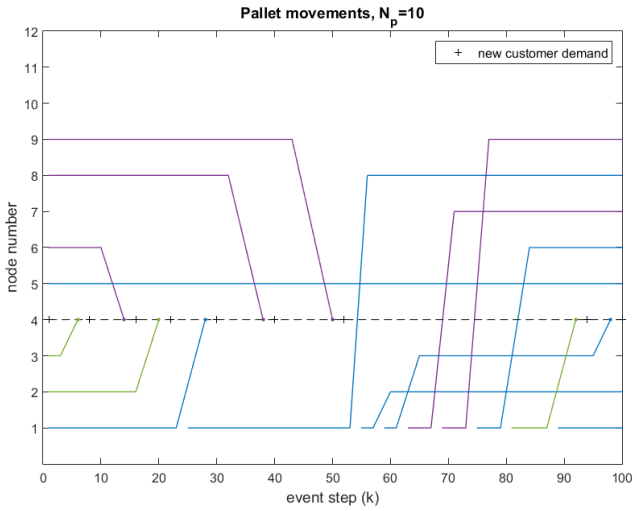

Fig. 3. Simulation results: movement of the pallets

The MLD model together with the objective function are translated into an integer linear programming problem. The control problem is implemented in the YALMIP (Löfberg, 2004) modelling language, and simulations have been performed.
Figure 3 shows the pallet movements conducted from simulation. The horizontal axis shows the events steps, and the vertical axis represents all the node numbers. The graph shows for every event step in which node a pallet is present. By connecting the positions of a pallet over time, the lines in the graph are created. Note that when a pallet is moved from one node to another, a linear line is depicted in the graph, even though the crane does not necessarily travel in a linear line. Each pallet type is represented by a different colour.
The graphs shows that at event step $k = 0$ there are

pallets present in nodes $N_1, N_2, N_3, N_5, N_6, N_8$, and N_9. The first pallet movement takes place from node N_3 to N_4, to satisfy the customer demand. It can also be noted that after a pallet is removed from source node N_1, this node is soon refilled with a new pallet that arrives from outside the system. Based on the type of pallet demanded by the customer at node N_4, the pallets present in the system, and the objective function, the control algorithm finds the optimal movements of the pallets through the system.

6. CONCLUSIONS

To control an AS/RS this work has proposed to use MLD modelling and propositional calculus to form an integer programming problem, which can be combined with MPC. To the best of our knowledge, the MLD-MPC method has never been applied before on an AS/RS. The case study shows that this control method works well for the laboratory stacker crane. A topic for future work involves improving the way of modelling by reducing the complexity of the model and decreasing the required computation time. The focus hereby should be on reduction of the number of integer variables, since these determine the complexity of an integer linear programming problem. The current model can easily be extended, e.g. to a form with multiple final storage nodes.
Next to these improvements and extensions it would be interesting to compare the MLD-MPC method to other control methods using MPC, such as time instant optimisation MPC (van Ekeren et al., 2013), and heuristics.

REFERENCES

Ávila, T., Corberán, A., Plana, I., and Sanchis, J. (2015). The stacker crane problem and the directed general routing problem. *Networks*, 65(1), 43–55.

Beccuti, A., Geyer, T., and Morari, M. (2005). A hybrid system approach to power systems voltage control. In *44th IEEE Conference on Decision and Control, and European Control Conference*, 6774–6779. Seville, Spain.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3), 407–427.

Bessenouci, H., Sari, Z., and Ghomri, L. (2012). Meta-heuristic based control of a flow rack automated storage retrieval system. *Journal of Intelligent Manufacturing*, 23(4), 1157–1166.

Camacho, E. and Bordons, C. (1997). *Model Predictive Control in the Process Industry*. Springer-Verlag New York, Inc.

Cataldo, A., Perizzato, A., and Scattolini, R. (2015). Production scheduling of parallel machines with model predictive control. *Control Engineering Practice*, 42, 28–40.

Cataldo, A. and Scattolini, R. (2014). Modeling and model predictive control of a de-manufacturing plant. In *IEEE Conference on Control Applications*, 1855–1860. Antibes, France.

Cavalier, T., Pardalos, P., and Soyster, A. (1999). Modeling and integer programming techniques applied to propositional calculus. *Computers & Operations Research*, 17(6), 561–570.

Dooly, D. and Lee, H. (2008). A shift-based sequencing method for twin-shuttle automated storage and retrieval systems. *IIE Transactions*, 40(6), 586–594.

Du, J., Song, C., and Li, P. (2009). Multilinear model control of hammerstein-like systems based on an included angle dividing method and the MLD-MPC strategy. *Industrial & Engineering Chemistry Research*, 48(8), 3934–3943.

Flegel, H. (2014). Manufuture, a vision for 2020. Technical report, European Commission. Manufuture High Level Group.

Gharehgozli, A., Yu, Y., Zhang, X., and De Koster, R. (2014). Polynomial time agorithms to minimize total travel time in a two-depot automated storage/retrieval system. *Transportation Science*.

Groot, N., De Schutter, B., and Hellendoorn, H. (2013). Integrated model predictive traffic and emission control using a piecewise-affine approach. *IEEE Transactions on Intelligent Transportation Systems*, 14(2), 587–598.

Han, M., McGinnis, L., Shieh, J., and White, J. (1987). On sequencing retrievals in an automated storage/retrieval system. *IIE Transactions*, 19(1), 55–66.

Lee, H. (1997). Performance analysis for automated storage and retrieval systems. *IIE Transactions*, 29(1), 15–28.

Löfberg, J. (2004). Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*. Taipei, Taiwan.

Mahajan, S., Rao, B., and Peters, B. (1998). A retrieval sequencing heuristic for miniload end-of-aisle automated storage/retrieval systems. *International Journal of Production Research*, 36(6), 1715–1731.

Qin, S. and Badgwell, T. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7), 733–764.

Raman, R. and Grossmann, I. (1992). Integration of logic and heuristic knowledge in MINLP optimization for process synthesis. *Computers & Chemical Engineering*, 16(3), 155–171.

Roodbergen, K. and Vis, I. (2009). A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, 194(2), 343–362.

van Ekeren, H., Negenborn, R., van Overloop, P., and De Schutter, B. (2013). Time-instant optimization for hybrid model predictive control of the Rhine-Meuse Delta. *Journal of Hydroinformatics*, 15(2), 271–292.

Vargas-Villamil, F. and Rivera, D. (2000). Multilayer optimization and scheduling using model predictive control: application to reentrant semiconductor manufacturing lines. *Computers & Chemical Engineering*, 24(8), 2009–2021.

Xi, Y., Li, D., and Lin, S. (2013). Model predictive control status and challenges. *Acta Automatica Sinica*, 39(3), 222–236.

Yu, Y. and De Koster, R. (2012). Sequencing heuristics for storing and retrieving unit loads in 3d compact automated warehousing systems. *IIE Transactions*, 44(2), 69–87.