MULTI-AGENT MODEL PREDICTIVE CONTROL FOR TRANSPORTATION NETWORKS WITH CONTINUOUS AND DISCRETE ELEMENTS

Rudy R. Negenborn * **Bart De Schutter** * **Hans Hellendoorn** *

* *Delft Center for Systems and Control*
Delft University of Technology,
Mekelweg 2, 2628 CD Delft, The Netherlands
{*r.r.negenborn,b.deschutter,j.hellendoorn@tudelft.nl*}

Abstract: For several reasons, control of transportation networks like road traffic networks, power transmission networks, water distribution networks, etc. has to be done using a multi-agent control approach. We consider a multi-agent control approach in which each control agent uses Model Predictive Control (MPC) to determine its actions. Each control agent uses a model of its subnetwork and communication with its neighboring control agents to come to agreement on the evolution of interconnections and to determine optimal local inputs and states. The challenge that is addressed in this paper comes from the situation that arises when the models that the agents use contain *both* continuous *and* discrete elements, instead of one of the two exclusively. We propose an approach that deals with this hybrid behavior and results in agreement between the agents with respect to the evolution of interconnections and that moreover provides integer inputs to obtain these evolutions.

Keywords: Multi-agent systems, model predictive control, transportation networks, hybrid systems.

1. INTRODUCTION

Transportation networks, like road traffic networks, power distribution networks, railway networks, water distribution networks, gas networks, sewer networks, etc. are usually large in size, consist of multiple subnetworks, have many actuators and sensors, and therefore show complex dynamics. These transportation networks can be considered at a generic level, at which commodity is brought into the network at sources, flows over links to sinks, and is influenced in its way of flowing by elements inside the network. The similarities between several types of transportation networks are the motivation for studying these networks in a generic way. Results obtained for generic transportation networks can then be specialized and fine-tuned for specific domains.

Transportation networks can often be considered as consisting of both continuous and discrete elements. E.g., commodity flows or speeds could be expressed in continuous variables, while control actions like traffic signal or speed limit settings are often expressed in discrete variables. Conventionally either only continuous or only discrete elements are considered. We consider both elements at the same time.

Control goals for transportation networks often involve avoiding congestion of links, minimizing costs of control actions, maximizing throughput, etc. For the type of networks that we consider centralized control, in which a single control agent determines the actions for the whole network, is often not feasible or impractical for several reasons. Some of these reasons are communication delays, too high computational requirements, unavailability of information from one

network operator to another, etc. Also, robustness and reliability of the network cannot be guaranteed when the single control agent breaks down.

For these reasons, employing a multi-agent, or distributed, control approach for control of these networks is a necessity (Bertsekas and Tsitsiklis, 1997; Siljak, 1991; Trave *et al.*, 1989). We consider a situation in which a division of the overall network into multiple smaller subnetworks is given. This is a situation that typically appears in practice, where different control authorities control different parts of the transportation network, e.g., the freeway network and the urban road networks are typically controlled by different road authorities.

Our starting point is a multi-agent Model Predictive Control (MPC) scheme that we introduced for control of transportation networks consisting of only continuous elements (Negenborn *et al.*, 2006). This scheme converges to an optimal overall solution in the case of a convex overall control problem. However, when also discrete variables are involved, e.g., as control actions, the overall optimization problem, i.e., the single control problem of controlling the whole network, is no longer convex. We now extend the scheme considered earlier to the situation where discrete variables play a role as local actions to the subnetworks, and where continuous variables only show up in interconnecting constraints between subnetworks. This situation appears, e.g., in transportation networks, when local actions consist of discrete speed limit settings and interconnecting constraints between subnetworks are expressed in terms of continuously modeled car flows.

This paper is organized as follows. In Section 2 we introduce the way in which we model transportation networks. In Section 3 we briefly discuss the multi-agent MPC scheme that we employ. In Section 4 we discuss the difficulties that arise when extending the multi-agent MPC scheme to the case involving both continuous and discrete variables and we propose an approach to deal with these difficulties. In Section 5 we give an example that indicates the workings of the proposed approach.

2. TRANSPORTATION NETWORK MODEL

Consider a transportation network partitioned into n subnetworks, each controlled by a control agent that has only a model of its own subnetwork. The interconnections between subnetworks are modeled using so-called *internetwork variables*. These variables express, e.g., what the continuous flow of cars between two subnetworks is. We distinguish two types of internetwork variables: internetwork *input* variables and internetwork *output* variables. On one side, the model of subnetwork i contains an internetwork input variable $w^{ji}_{\text{in},k} \in \mathbb{R}^{n_{w,\text{in},ji}}$ that represents the input caused by subnetwork j on the state of subnetwork i at time step k. On the other hand, the model of subnetwork

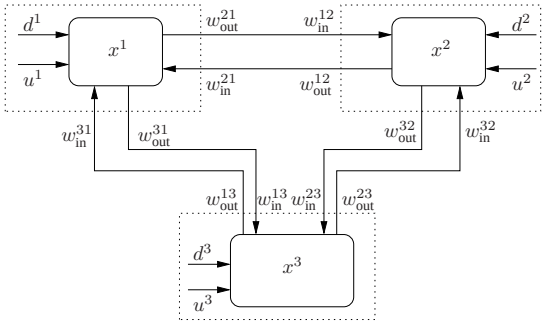

Fig. 1. Each subnetwork model has a set of variables. Internetwork variables form interconnecting constraints between variables of two subnetwork.

j contains an internetwork output variable $w^{ij}_{\text{out},k} \in \mathbb{R}^{n_{w,\text{out},ij}}$ that represents the influence that subnetwork j has on subnetwork i. In the physical network the internetwork input to subnetwork i from j must be equal to the internetwork output from subnetwork j to i, e.g., the commodity flow going out from subnetwork j into subnetwork i has to equal the commodity flow going into subnetwork i coming out from subnetwork j. This means that so-called interconnecting constraints have to be satisfied between subnetwork i and its neighboring subnetworks $j \in N_i$, (where $N_i = \{j^i_1, \ldots, j^i_{m_i}\}$ is the set of indexes of the m_i subnetworks connected to subnetwork i), i.e.,

$$w^{ji}_{\text{in},k} = w^{ij}_{\text{out},k}, \qquad w^{ji}_{\text{out},k} = w^{ij}_{\text{in},k}, \quad \forall j \in N_i, \quad (1)$$

at all time steps k, as illustrated in Fig. 1.

We model the dynamics of subnetwork i by a linear discrete-time difference equation:

$$x^i_{k+1} = A^i x^i_k + B^i_1 u^i_k + B^i_2 d^i_k + B^{j_1 i}_3 w^{j_1 i}_{\text{in},k} + \ldots + B^{j_m i}_3 w^{j_m i}_{\text{in},k} \quad (2)$$

$$w^{j_1 i}_{\text{out},k} = C^{j_1 i}_{\text{out}} x^i_k, \quad (3)$$

$$\vdots \qquad \vdots$$

$$w^{j_m i}_{\text{out},k} = C^{j_m i}_{\text{out}} x^i_k, \quad (4)$$

where at time step k, for subnetwork i, $x^i_k \in \mathbb{R}^{n_{x,i}}$ are dynamic states, $u^i_k \in [u^i_{\min}, \ldots, u^i_{\max}]$ are discrete inputs that take on discrete values of the interval from $u^i_{\min}$ until $u^i_{\max}$, $d^i_k \in \mathbb{R}^{n_{d,i}}$ are local disturbances, and $w^{ji}_{\text{in},k}$ and $w^{ji}_{\text{out},k}$ are the continuous internetwork input and output variables respectively. The matrices A^i, B^i_1, B^i_2, B^i_3 are of appropriate dimensions and determine how the different variables influence the state of subnetwork i, while the matrix C^{ji}_{out} is of appropriate dimensions and contains 0 entries on each row, except for a single 1 corresponding to a variable that is an internetwork output of subnetwork i with respect to subnetwork j.

Note that the inputs u^i_k are limited to discrete values. There are two different types of discrete inputs: discrete inputs that have a direct meaning as a quantity since they are represented as numbers, and discrete inputs that only have a symbolic meaning. E.g., in the former class of discrete inputs we have discrete

numbers of cars that are allowed to enter a road; in the latter class we have, e.g., the switching of traffic signals from red to green. Here we are interested in the first class of discrete inputs. Note that however the second class of discrete inputs can sometimes be transformed into the first class of inputs.

Note moreover that with the inputs u_k^i we refer to the actions of controllers, and not the physical inputs at the border of the network. The physical inputs to the network and the physical demands leaving the network at the border are modeled as disturbances d_k^i.

In the following the elements of x_k^i, u_k^i, and d_k^i are referred to as *local* variables z_k^i of agent i, while the variables $w_{\text{in},k}^{ji}$ and $w_{\text{out},k}^{ji}$ together are referred to as *internetwork* variables w_k^{ji} of agent i.

3. MULTI-AGENT MPC

From here on, we assume that the network has been divided into subnetworks, each subnetwork has been assigned an agent, and each agent has a subnetwork model (2)–(4) of its own subnetwork.

To determine which actions to take each agent uses an MPC scheme (Maciejowski, 2002; Morari and Lee, 1999). Advantages of MPC are its explicit way of dealing with constraints and its easy way of integrating forecasts. For transportation networks MPC provides a convenient way to include, e.g., capacity limits on links, maximums on queue lengths, measurements from upstream sensors, profiles of demands, etc.

In MPC, each agent has to determine inputs u_k^i to its actuators that give good local and overall performance over a horizon of N steps according to an objective function $J_z^i(\tilde{z}_k^i)$, where the symbol $\tilde{}$ over a variable indicates variables over the horizon, e.g., $\tilde{z}_k^i = [(z_k^i)^T \ \dots \ (z_{k+N-1}^i)^T]^T$. In this paper we take the local cost term to have a quadratic structure, i.e., $J_z^i(\tilde{z}_k^i) = (\tilde{z}_k^i)^T Q_z^i \tilde{z}_k^i$, where Q_z^i is a weighting matrix, although the following can also be extended to the case where we would have a tracking cost term of the form $J_z^i(\tilde{z}_k^i) = (\tilde{z}_k^i - \tilde{z}_{\text{ref},k}^i)^T Q_z^i (\tilde{z}_k^i - \tilde{z}_{\text{ref},k}^i)$.

Since the subnetwork model that each agent has depends on neighboring subnetwork models through the internetwork input and output variables and the interconnecting constraints, the agents have to use a negotiation scheme to obtain agreement on how these variables should evolve over the prediction horizon. Using such a scheme, the agents reduce the uncertainty in making predictions of the evolution of the variables of their own subnetwork model. To obtain agreement, the agents perform at each decision step a series of iterations of solving the following problem:

$$\min_{\tilde{z}_k^i, \tilde{w}_k^i} \ J_z^i(\tilde{z}_k^i) + \sum_{j \in N_i} J_w^i(\tilde{w}_k^{ji}), \tag{5}$$

subject to the dynamics of the subnetwork (2)–(4) over the horizon, where the additional cost term $J_w^i(\tilde{w}_k^{ji})$ deals with the internetwork variables. The particular structure of this additional cost term depends on the negotiation scheme used. Its shape is adjusted at the end of each iteration in such a way that at the end of the iterations, ideally, the interconnecting constraints (1) are satisfied and moreover the actions that the agents choose are overall optimal.

The multi-agent MPC scheme that we employ is a serial scheme that is the result of using a *Block Coordinate Descent* (Bertsekas and Tsitsiklis, 1997; Royo, 2001) to decompose an Augmented Lagrangian formulation (Bertsekas, 2003) of the control problem at hand. The scheme is serial in the sense that one agent after another minimizes its problem (5) to determine its optimal local and internetwork variables, while the variables of the other agents stay fixed. When the agents have reached agreement on the internetwork variables after a number of iterations, the iterations terminate and the determined actions are implemented. We use this serial implementation since in earlier studies this scheme showed to outperform a parallel implementation in terms of convergence speed and required computations (Negenborn *et al.*, 2006)

The serial implementation uses the following structure for the additional cost term $J_w^i(\tilde{w}_k^{ji})$:

$$J_w^i(\tilde{w}_k^{ji}) = \begin{bmatrix} (\tilde{\lambda}_s^{ji})^T & (-\tilde{\lambda}_s^{ij})^T \end{bmatrix} \begin{bmatrix} \tilde{w}_{\text{in},k}^{ji} \\ \tilde{w}_{\text{out},k}^{ji} \end{bmatrix} + \frac{c}{2} \left\| \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \tilde{w}_{\text{in, prev},k}^{ij} \\ \tilde{w}_{\text{out, prev},k}^{ij} \end{bmatrix} - \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \begin{bmatrix} \tilde{w}_{\text{in},k}^{ji} \\ \tilde{w}_{\text{out},k}^{ji} \end{bmatrix} \right\|^2,$$

where given are:

- the information $\tilde{w}_{\text{prev},k}^{ij} = \tilde{w}_{s+1|k}^{ij}$ computed at the current iteration s for each agent $j \in N_i$ that has solved its problem *before* agent i in the *current* iteration s,
- the information $\tilde{w}_{\text{prev},k}^{ij} = \tilde{w}_{s|k}^{ij}$ computed at the *previous* iteration for the other agents,

and where c is a positive scalar penalizing interconnecting constraint violations, and where the λs are Lagrangian multipliers, updated at the end of each iteration to encourage convergence to internetwork variables that satisfy the interconnecting constraints.

The serial multi-agent MPC implementation can now be outlined as follows: at decision step k, iteration s:

(1) For $i = 1, \dots, n$, one agent after another:
   (a) Agent i determines $\tilde{z}_{s+1|k}^i, \tilde{w}_{s+1|k}^{ji}$ by solving (5).
   (b) Agent i sends to agent $j \in N_i$ the computed values $\tilde{w}_{s+1|k}^{ji}$.

(2) After all agents have solved their problems at one iteration, they update their Lagrangian multipliers using:

$$\tilde{\lambda}_{s+1}^{ji} = \tilde{\lambda}_s^{ji} + c(\tilde{w}_{\text{in},s+1|k}^{ji} - \tilde{w}_{\text{out},s+1|k}^{ij}) \tag{6}$$

$$\tilde{\lambda}_{s+1}^{ij} = \tilde{\lambda}_s^{ij} + c(\tilde{w}_{\text{in},s+1|k}^{ij} - \tilde{w}_{\text{out},s+1|k}^{ji}). \tag{7}$$

(3) Each agent moves to the next iteration $s+1$, and the cycle starts over, unless the infinity norm of the terms $\tilde{w}^{ji}_{\text{in},s+1|k} - \tilde{w}^{ij}_{\text{out},s+1|k}$ is below a small positive threshold ϵ.

The λ variables can be initialized arbitrarily, although choosing them closer to optimal λ^* yields improved performance in terms of convergence speed. Therefore, initializing the Lagrangian multipliers with values computed at the previous decision step yields improved performance in terms of number of iterations required.

Depending on the problem at hand, convergence will or will not appear. For the case of an overall convex problem, thus involving only continuous variables, this scheme will indeed converge to an overall optimal solution (Negenborn *et al.*, 2006), or at least as optimal as indicated by the magnitude of ϵ. The closer ϵ is to zero, the closer the found solution is to the overall optimal solution, since at the overall optimal solution the update (6)–(7) will not change the Lagrangian multipliers anymore.

However, when discrete inputs appear in the control problem, the overall control problem is no longer convex and measures have to be taken to guarantee that the iterations terminate with at least a feasible solution, and even better with a solution that is close to or equal to an overall optimal solution. In the next section we propose an approach to this problem.

4. DEALING WITH DISCRETE VARIABLES

We consider the case where the local actions to the subnetworks are modeled with integer variables that take on values from the finite set $\{u^i_{\min}, \ldots, u^i_{\max}\}$. When discrete elements appear in the optimization problem, the optimization problem (5) for each agent i can be formulated as a mixed-integer quadratic programming problem. Due to the discrete elements, the convexity assumption that guarantees that the scheme discussed in the previous section converges to an overall optimal solution does not hold anymore.

In this situation it may be the case that the agents cannot come to agreement on the internetwork variables, while choosing locally optimal integer inputs. So, a periodic sequence might arise of agents making suggestions for values of the internetwork variables that not all other agents agree with. This periodic sequence will continue without converging.

In practice at some point a decision has to be made on which actions to implement. A common approach to deal with integer inputs is to relax them to continuous variables and at the end of the optimization round them to the closest integer value. In particular when making predictions over a longer horizon this rounding can cause at least sub-optimality and sometimes even infeasibility. This is due to the fact that in general a rounded input has a different influence on the evolution of the network over a time step then a continuous input would have. So in practice the evolution of the network will be different than the evolution used in the optimization. In the following we discuss a number of alternative approaches, ultimately leading to an approach that does not relax the integer inputs to continuous variables and that moreover ensures that the evolution of the network as used in the optimization is the same as the evolution that will be encountered in practice.

That the iterations continue means that the stopping criterion is not met. We discuss 5 ways to deal with this and to make the iterations stop:

1. Accuracy threshold increments. The accuracy threshold ϵ is used in the stopping criterion to determine when the iterations should stop. If this threshold is increased, the iterations will sooner stop. However, this of course reduces the quality of the determined solutions. In addition, ignoring the violations of the interconnecting constraints can obviously lead to sub-optimally chosen inputs and since no agreement has been achieved on the values of the internetwork variables the predictions that each agent has over the evolution of its subnetwork will be inaccurate.

2. Discretization refinements. By making the discretization of the inputs finer, at some point the discretization will be fine enough to let the Lagrangian multipliers converge to values that make the stopping condition satisfied. For a specific tolerance ϵ there is a certain minimum discretization at which the iterations converge to Lagrangian multipliers that satisfy the stopping condition. If a coarser discretization is chosen, then the periodic behavior could emerge. In practice, however, the discretization of the inputs may be given and may not be adjustable.

3. Addition of continuous dummy inputs. Continuous dummy inputs can be included in the optimization problems to compensate constraint violations of the interconnecting constraints by providing the remaining part of the desired input that the integer input cannot provide. A very high cost will be associated with this type of input such that it will only be used if it is really not worth changing the integer inputs. Due to the dummy inputs the agents can obtain agreement on the internetwork variables. The dummy inputs indicate how much more or less control action the agent has to provide in order to fulfill the agreements that an agent made.

4. Penalty term increments. The penalty term c can be increased to a very high value once the periodic behavior has been detected, which can be done by either waiting a minimum number of iterations or by monitoring the sequence of solutions. In this way the interconnecting constraints are forced to be satisfied, and the inputs that come with this can be implemented. However, by the time that the periodic behav-

ior emerges, the agents have already determined certain local values for their local variables. By imposing a very high c, it will still take a significant number of iterations before the agents obtained values that do make the interconnecting constraints satisfied.

5. Integer input fixations. The integer inputs can be fixed once the periodic behavior has been detected. E.g., the integer inputs can be fixed to the locally most optimal integers, or they can be fixed to the most frequently appearing value for the integer inputs over the last periodic cycle. The remaining overall optimization problem will become convex and the iterations will converge to optimal values for the remaining variables. The remaining variables will be optimal with respect to the fixed setting of the integer inputs, which however may be sub-optimal from a network-wide perspective. More importantly though, at the end of the iterations the interconnecting constraints will be satisfied and thus the agents will have agreed on how the internetwork variables should evolve over the prediction horizon. Furthermore, they will have determined inputs that ensure that this agreement is fulfilled within a reasonable number of iterations (contrarily to the previous case).

In the following section we experimentally assess the performance of this last scheme. We consider this last scheme, since this is the only scheme that both guarantees agreement between the agents within a reasonable number of iterations and provides feasible integer inputs. The scheme has the following outline:

(1) Start as in the continuous case with a fixed, low, value for c, e.g., 1 or 10, arbitrarily initialized Lagrangian multipliers, and, e.g., $\epsilon = 0.00001$.
(2) If the iterations converge to a fixed solution, then the interconnecting constraints will be satisfied and the resulting integer inputs will be optimal.
(3) Otherwise, if a periodic switching between solutions of different agents is detected, the integer inputs are fixed to the value that appeared most frequently over the last periodic cycle. After this, the iterations continue as before, but with fixed integer inputs.

By fixing the integer inputs, the optimization problems become convex and the subsequent iterations will bring the internetwork variables to values that make the interconnecting constraints satisfied with feasible, although not necessarily optimal, integer inputs.

5. EXAMPLE

5.1 Setup

We consider a general transportation network divided into two subnetworks, each being controlled by a control agent, see Fig. 2. Through the network there is a flow of a continuous commodity, e.g., a flow of cars. The two subnetwork are connected to each

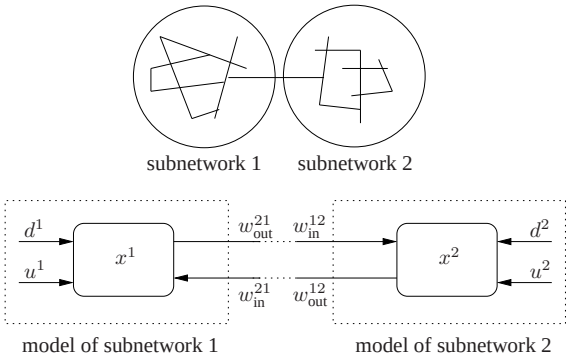

Fig. 2. Illustration of an abstract transportation network consisting of two subnetworks (top), and the variables of the subnetwork models considered by the 2 control agents (bottom).

other through an interconnecting link, e.g., a shared road or highway. Each subnetwork $i \in \{1, 2\}$ has as control input a controllable source or sink, that generates or consumes commodity u^i_k in discrete steps respectively. E.g., an on-ramp metering installation at the border of each subnetwork with sufficiently high demand represents a source, while a forced queuing of cars represents at some times a sink, while at others a source. In addition, each subnetwork has as local disturbance an unexpected addition or removal of commodity, d^i_k, e.g., a flow of cars that come and go to a car park or a shopping mall outside the control of the control agents. The state of subnetwork i is denoted by x^i_k and may encode, e.g., the average number of cars in each part of the subnetworks over a time step. We assume that the dynamics of the subnetworks can be modeled by (2)–(4). See for more details of such a model (Negenborn *et al.*, 2006).

In order to make good local decisions, the agents have to obtain agreement on the commodity flowing over the line from subnetwork i to j. To compute this flow elements of both the state x^i_k and the state x^j_k have to be known. These elements are the internetwork variables that form the interconnecting constraints between the control problems of both agents.

Assume in subnetwork i a sudden commodity injection. In order to prevent the subnetwork from ending up in an undesirable state, e.g., a state with traffic congestion, the agents controlling the subnetworks have to come to agreement on how much commodity should be transported over the interconnecting link.

5.2 Results

Figure 3 shows a typical evolution of the constraint violation of the interconnecting constraint associated with a prediction step. This constraint violation is simply the difference between the internetwork input variable of one agent and the corresponding internetwork output variable of the other agent, and can represent, e.g., the mismatch between the commodity flow that each of the two agents wants to send over

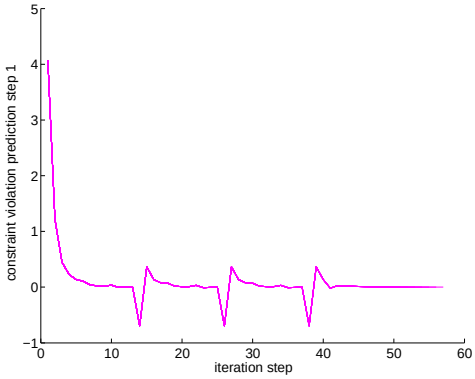

Fig. 3. Evolution of the constraint violation for the interconnecting constraint associated with the first prediction step. At iteration step 40, the integer input fixation function is activated.

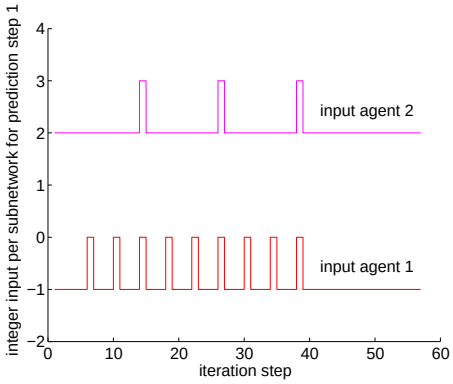

Fig. 4. Evolution of the integer inputs as computed by the two agents over the iterations. From iteration step 40, the integer inputs are kept fixed.

the interconnecting link. As can be seen, the agents are not able to come to agreement on values for the interconnecting constraints over the first 40 steps. This is also seen in Figure 4, which shows the integer inputs as computed by the two agents for the first prediction step. We clearly see that the inputs do not converge to an optimal input, but keep switching. At iteration step 40 the input fixation function is activated and the integer inputs are kept fixed at the values that appeared most frequently over the last periodic cycle. Once the integer inputs have been fixed the agents quickly obtain agreement on the interconnecting constraint.

6. CONCLUSIONS & FUTURE RESEARCH

In this paper we have considered control of transportation networks, like road traffic networks, power networks, water distribution networks, and so on using multi-agent Model Predictive Control (MPC), to be used when a single-agent approach is inapplicable. In this setting, the network is divided into a number of subnetworks, each being controlled by a control agent that uses a model of its subnetwork and MPC to determine its actions. We have focused on approaches that deal with subnetwork models that contain both continuous and discrete elements, contrarily to conventional approaches where only either one of the two is considered. In our setup, discrete elements appear in the form of discrete control actions, while continuous elements appear due to, e.g., continuous flows of commodity over the network.

We have started with a serial multi-agent MPC scheme that converges to optimality for convex overall optimization problems, involving only continuous variables. We have pointed out why this scheme may not converge (and therefore not stop) when also discrete variables are included. We have discussed a number of approaches that could be considered to make the scheme terminate. We have chosen an approach that may give sub-optimal integer inputs, but that will ensure that the interconnecting constraints between subnetworks are satisfied. An example confirms this.

Future research lies in determining how large the suboptimality of the resulting solutions is compared to a hypothetical centralized controller that has access to all relevant information. Moreover, the detection of when periodic solutions start appearing can be improved to reduce the number of iterations required before termination. We will also perform experiments on transportation networks of more realistic size to further assess the potential of the proposed approach. And finally, we will investigate in which situations a multi-agent approach is more efficient than a single-agent approach.

ACKNOWLEDGMENTS

Research supported by STW project “Multi-agent control of large-scale hybrid systems” (DWV.6188), Transport Research Centre Delft, and European Network of Excellence “HYbrid CONtrol: Taming Heterogeneity and Complexity of Networked Embedded Systems (HYCON)”.

REFERENCES

Bertsekas, D. P. (2003). *Nonlinear Programming*. Athena Scientific. Belmont, Massachusetts.

Bertsekas, D. P. and J. N. Tsitsiklis (1997). *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific. Nashua, New Hampshire.

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Prentice Hall. Harlow, England.

Morari, M. and J. H. Lee (1999). Model predictive control: past, present and future. *Computers and Chemical Engineering* **23**, 667–682.

Negenborn, R.R., B. De Schutter and J. Hellendoorn (2006). Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. In: *Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing*. Saint-Etienne, France.

Royo, C. B. (2001). Generalized Unit Commitment by the Radar Multiplier Method. PhD thesis. Technical University of Catalonia. Barcelona, Spain.

Siljak, D. D. (1991). *Decentralized Control of Complex Systems*. Academic Press. Boston.

Trave, L., A. Titli and A. Tarras (1989). *Large Scale Systems: Decentralization, Structure Constraints and Fixed Modes*. Springer-Verlag. London, UK.