Delft University of Technology

Delft Center for Systems and Control

Technical report 11-001

Fast model predictive control for urban road networks via MILP*

S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn

If you want to cite this report, please use the following reference instead:
S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "Fast model predictive control for urban road networks via MILP," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 846–856, Sept. 2011.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/11_001.html`

Fast Model Predictive Control for Urban Road Networks via MILP

Shu Lin, *Student Member, IEEE,* Bart De Schutter, *Senior Member, IEEE,* Yugeng Xi, *Senior Member, IEEE,* and Hans Hellendoorn

***Abstract*—In this paper, an advanced control strategy, i.e. Model Predictive Control (MPC), is applied to control and coordinate urban traffic networks. However, due to the nonlinearity of the prediction model, the optimization of MPC is a nonlinear non-convex optimization problem. In this case, the on-line computational complexity becomes a big challenge for the MPC controller, if it is implemented in real-life traffic network. To overcome this problem, the on-line optimization problem is reformulated into a Mixed-Integer Linear Programming (MILP) optimization problem, so as to increase the real-time feasibility of the MPC control strategy. The new optimization problem can be solved very efficiently by existing MILP solvers, and the global optimum of the problem is guaranteed. Moreover, we propose an approach to reduce the complexity of the MILP optimization problem even further. The simulation results show that the MILP-based MPC controllers can reach the same performance, but the time taken to solve the optimization becomes only a few seconds, which is a significant reduction compared with the time required by the original MPC controller.**

***Index Terms*—Urban traffic network control, Model predictive control, Urban traffic modeling.**

I. Introduction

TRAFFIC control is one of the most efficient and also effective ways to reduce traffic congestion and to alleviate the problems caused by congestion. To reduce traffic congestion from a network-wide point of view, advanced traffic control strategies are needed to coordinate traffic control measures throughout traffic networks.

There are already well-known coordinated traffic-responsive control strategies for urban traffic networks, SCOOT [1], [2] and SCATS [3], which are widely used in many cities around the world [4]. They are both dynamic traffic control strategies based on measured current traffic states in distributed, multi-level, hierarchical system structures. It has already been shown that these two systems work effectively in real traffic world. But, these two systems are more focusing on dynamic intersection controllers, and local coordinations that consider only a few neighbor intersections. In the 1980s and 1990s, a number of model-based optimization control strategies based on simple traffic models emerged, e.g. OPAC [5], PRODYN [6], CRONOS [7], RHODES [8], and MOTION [9], which can forecast the future traffic behavior of the network based on models. With these forecasting models, the control strategies are able to make control decisions to guarantee better performance within an area of the traffic network in a near future. However, the models used in these control approaches are mainly simple traffic models based on the traffic data measured by upstream detectors, which to some extent limits the performance for the future. Moreover, the on-line computational complexity of these strategies is a practical issue that will be encountered when implemented in urban traffic networks [4]. Coordinated traffic-responsive control strategies that are able to avoid parts of the on-line computational complexity, were also proposed. UTOPIA/SPOT [10] is a hierarchical system with simple local intersection controllers and a central controller for an area of urban networks. The central controller optimizes the control actions for the whole area based on the model of the network. The local controller makes the decision only based on local information, but with a penalty term to guarantee that the local decision is not too far from the central decision. Therefore, UTOPIA/SPOT avoids part of the on-line computational burden, but results in suboptimal solutions. TUC [11], [12] was proposed for controlling an urban traffic network based on the well-known simple store-and-forward model. TUC designs a feedback regulator off-line based on the store-and-forward model, and on-line derives the traffic signals using a feedback control law by feeding it with the real-time measured traffic states. Therefore, the TUC strategy reduces the on-line computational complexity significantly by moving the time-consuming optimization off-line. However, when the real traffic conditions change, the feedback control law needs to be redesigned according to the new current traffic conditions, which is also computational complex if it occurs too frequently. Moreover, more urban traffic control and simulation systems are also proposed recently [13], [14].

In recent years, some macroscopic urban traffic models that can describe the traffic dynamics of the whole urban traffic network, have been developed. Subsequently, model-based optimization control strategies (including Model Predictive Control (MPC)) [15]–[17] based on these prediction models have been developed. The framework of model-based predictive control strategies contains three classical steps: prediction, on-line optimization, and rolling time horizon. Due to this framework, model-based optimization control methods are capable of coordinating all the traffic signals of intersections within the urban traffic networks; they are robust to the uncertainty, disturbances, and even the model mismatch in reality; they are able to predict the future and thus avoid making myopic control decisions. All these advantages make

S. Lin, B. De Schutter, and J. Hellendoorn are with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands. {s.lin, b.deschutter, j.hellendoorn}@tudelft.nl

S. Lin and Y. Xi are with Department of Automation, Shanghai Jiao Tong University, and Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai, China. lisashulin@gmail.com, ygxi@sjtu.edu.cn

Manuscript received July 23, 2010

the model-based control methods very attractive. However, the biggest challenge for implementing a model-based optimization control strategy is the on-line computational complexity. The high on-line computational requirement almost makes them real-time infeasible[1] in the real-life implementation.

In order to improve the applicability of these strategies, some methods can be developed to solve or alleviate the computational problem: (i) dividing the network into small subnetworks, and building distributed controllers [18]–[22]; (ii) solving the optimization problem off-line, such as optimizing a feedback regulator off-line and using it with real-time measured traffic states to derive control decisions [4], [12]; (iii) reducing the prediction model to make it more computational efficient, so as to improve the real-time feasibility of the optimization; (iv) approximating the optimization problem by one that can be solved more computationally efficient. In this paper, we mainly focus on improving the real-time feasibility of MPC controllers through improving the efficiency of solving optimization problems.

This paper is organized as follows. In Section II, a macroscopic urban traffic model (S model) is described. In Section III, MPC controllers are designed for urban traffic networks based on the S model. Then the S model is reformulated in Section IV, and MILP-MPC controllers are established based on the reformulated model in Section V. The derived MILP problem is further reduced in Section VI. Section VII presents the simulation results, and Section VIII concludes the paper.

II. Macroscopic Urban Traffic Model (S model)

Considering the computational complexity, typically macroscopic models, which focus on traffic flows but not on individual vehicles, are selected as prediction models for MPC controllers. Since the on-line feasibility of the MPC controller is very important, the prediction model needs to provide a good trade-off between the accuracy and the computational complexity. A macroscopic simplified urban traffic model (S model) was proposed in [23] based on previous work [24]–[26]. It is demonstrated with experiments that this simplified traffic model reduces the simulation time significantly, compared with the model in [25], [26], with only a limited reduction in accuracy. Therefore, we also use this model here.

In the S model [23], we define J as the set of nodes (intersections), and L as the set of links (roads) in the urban traffic network. Link (u,d) is marked by its upstream node u ($u \in J$) and downstream node d ($d \in J$). The input and output links of link (u,d) can be also specified by the upstream and downstream nodes. The sets of input and output nodes for link (u,d) are $I_{u,d} \subset J$ and $O_{u,d} \subset J$.

In order to describe the evolution of the models, we first define some variables (see also Fig. 1):

[1]Real-time feasibility means that the on-line optimization problem can be solved fast enough so that the result is found before the time at which the controller should generate the next control signal.

$I_{u,d}$: set of input nodes of link (u,d),
$O_{u,d}$: set of output nodes of link (u,d),
k : simulation step counter,
$n_{u,d}(k)$ (veh) : number of vehicles in link (u,d) at step k,
$q_{u,d}(k)$ (veh) : queue length at step k in link (u,d), $q_{u,d,o}(k)$ is the queue length of the sub-stream turning to link o,
$\alpha^{\text{leave}}_{u,d}(k)$ (veh/h) : flow rate leaving link (u,d) at step k, $\alpha^{\text{leave}}_{u,d,o}(k)$ is the leaving flow rate of the sub-stream towards o,
$\alpha^{\text{arriv}}_{u,d}(k)$ (veh/h) : flow rate arriving at the end of the queue in link (u,d) at step k, $\alpha^{\text{arriv}}_{u,d,o}(k)$ is the arriving flow rate of the sub-stream towards o,
$\alpha^{\text{enter}}_{u,d}(k)$ (veh/h) : flow rate entering link (u,d) at step k, $\alpha^{\text{enter}}_{i,u,d}(k)$ is the flow rate entering link (u,d) from i,
$\beta_{u,d,o}(k)$: relative fraction of the vehicles in link (u,d) turning to o at step k,
$\mu_{u,d}$ (veh/h) : saturated flow rate leaving link (u,d),
$g_{u,d,o}(k)$ (s) : green time length during step k for the traffic stream towards o in link (u,d),
$v^{\text{free}}_{u,d}$ (km/h) : free-flow vehicle speed in link (u,d),
$C_{u,d}$ (veh) : capacity of link (u,d) expressed in number of vehicles,
$N^{\text{lane}}_{u,d}$: number of lanes in link (u,d),
$\Delta c_{u,d}$ (s) : offset between node u and node d, i.e. the offset time between the cycle times of the upstream and the downstream intersections at the beginning of every control time step,
l_{veh} (m) : average vehicle length.

In the S model, every intersection takes the cycle time as its simulation time interval. The cycle times for intersection u and d, which are denoted by c_u and c_d respectively, can be different from each other. So, in this situation, the simulation step counters of different intersections are not same. Moreover, as cycle times are the simulation time intervals, the input and output flow rates of the link are averaged over the cycle times in the S model.

Taking the cycle time c_d as the length of the simulation time interval for link (u,d) and k_d as the corresponding time step counter, the number of the vehicles in link (u,d) is updated as follows:

$$n_{u,d}(k_d+1) = n_{u,d}(k_d) + \left(\alpha^{\text{enter}}_{u,d}(k_d) - \sum_{o \in O_{u,d}} \alpha^{\text{leave}}_{u,d,o}(k_d) \right) \cdot c_d \ . \tag{1}$$

The leaving average flow rate over c_d is determined by

$$\begin{aligned} \alpha^{\text{leave}}_{u,d,o}(k_d) = \min\Big(& \beta_{u,d,o}(k_d) \cdot \mu_{u,d} \cdot g_{u,d,o}(k_d)/c_d, \\ & q_{u,d,o}(k_d)/c_d + \alpha^{\text{arriv}}_{u,d,o}(k_d), \\ & \frac{\beta_{u,d,o}(k_d)}{\sum_{u \in I_{d,o}} \beta_{u,d,o}(k_d)} \cdot \frac{C_{d,o} - n_{d,o}(k_d)}{c_d} \Big) \ . \end{aligned} \tag{2}$$

The number of vehicles waiting in the queue turning to link

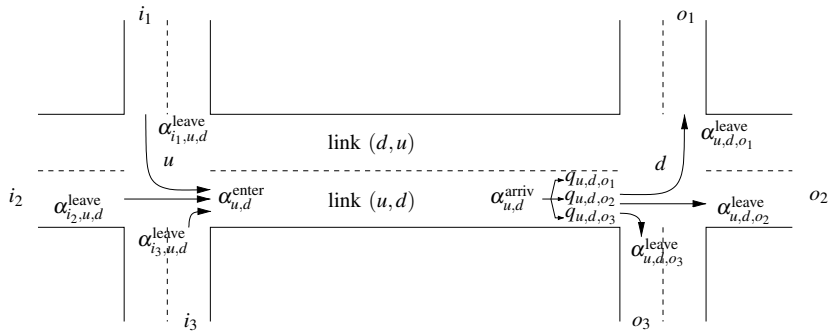


Fig. 1. A link connecting two traffic-signal-controlled intersections

o is updated as

$$q_{u,d,o}(k_d+1) = q_{u,d,o}(k_d) + \left(\alpha^{\text{arriv}}_{u,d,o}(k_d) - \alpha^{\text{leave}}_{u,d,o}(k_d)\right)\cdot c_d \quad . \tag{3}$$

The flow rate of vehicles that entered link (u,d) is given by

$$\alpha^{\text{arriv}}_{u,d}(k_d) = \frac{c_d-\gamma(k_d)}{c_d}\cdot\alpha^{\text{enter}}_{u,d}\left(k_d-\tau(k_d)\right) + \frac{\gamma(k_d)}{c_d}\cdot\alpha^{\text{enter}}_{u,d}\left(k_d-\tau(k_d)-1\right), \tag{4}$$

$$\tau(k_d) = \text{floor}\left\{\frac{\left(C_{u,d}-q_{u,d}(k_d)\right)\cdot l_{\text{veh}}}{N^{\text{lane}}_{u,d}\cdot v^{\text{free}}_{u,d}\cdot c_d}\right\},$$
$$\gamma(k_d) = \text{rem}\left\{\frac{\left(C_{u,d}-q_{u,d}(k_d)\right)\cdot l_{\text{veh}}}{N^{\text{lane}}_{u,d}\cdot v^{\text{free}}_{u,d}\cdot c_d}\right\}. \tag{5}$$

with floor$\{x\}$ referring to the largest integer smaller than or equal to x, and rem$\{x\}$ is the remainder.

Before reaching the tail of the waiting queues in link (u,d), the flow rate of arriving vehicles will be divided according to the turning rates:

$$\alpha^{\text{arriv}}_{u,d,o}(k_d) = \beta_{u,d,o}(k_d)\cdot\alpha^{\text{arriv}}_{u,d}(k_d). \tag{6}$$

As the simulation time intervals for the upstream and the downstream intersections are different from each other, the flow rates leaving from the upstream links and the flow rates entering the downstream links have to be synchronized as follows (see [27] for details).

As Fig. 2 illustrates, first, we define continuous-time flow rates leaving upstream links as

$$\alpha^{\text{leave,cont}}_{i,u,d}(t) = \alpha^{\text{leave}}_{i,u,d}(k_u), \quad k_u\cdot c_u \le t < (k_u+1)\cdot c_u, \tag{7}$$

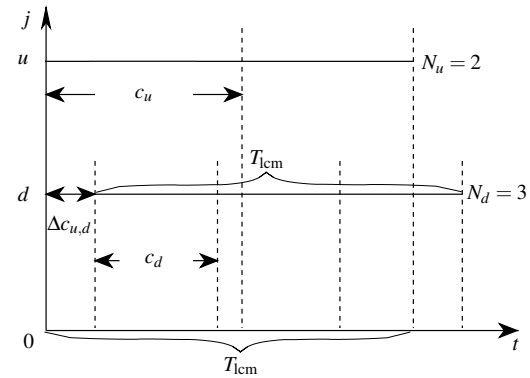
and then we discrete them for downstream links as

$$\alpha^{\text{enter}}_{i,u,d}(k_d) = \frac{1}{c_d}\int_{k_d\cdot c_d+\Delta c_{u,d}}^{(k_d+1)\cdot c_d+\Delta c_{u,d}} \alpha^{\text{leave,cont}}_{i,u,d}(t)\,\mathrm{d}t \quad . \tag{8}$$

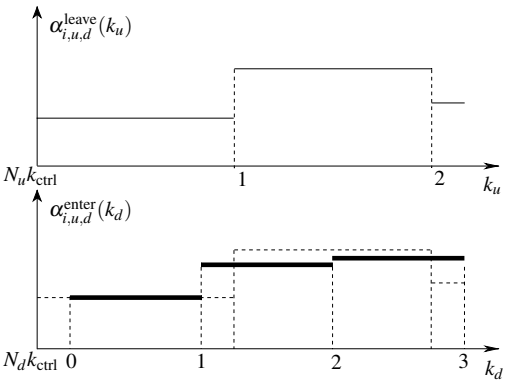
III. Model-based Urban Traffic Network Control

Model Predictive Control (MPC) [28] is a methodology that repeatedly solves optimization problems on-line in a rolling horizon way to derive a sequence of control decisions.

A common control time interval T_{ctrl} is defined as $T_{\text{ctrl}} = N\cdot T_{\text{lcm}}$, , where N is an integer, so that intersections can communicate with each other and be synchronous. For any given



(a) Relationship between cycle times and control time interval



(b) Illustration for synchronizing flow rates

Fig. 2. Synchronization of upstream and downstream intersections

model simulation time step counter k_d of intersection $d \in J$, the corresponding value of k_{ctrl} is given by $k_{\text{ctrl}}(k_d) = \text{floor}\left\{\frac{k_d c_d}{T_{\text{ctrl}}}\right\}$, where floor$\{x\}$ is the largest integer smaller than or equal to x.

The MPC approach can be described by the following three steps:

1) **Prediction model**. The S model presented in Section II above can be used as prediction model for the MPC controller. It can be generally described, for all $k_d c_d \in [k_{\text{ctrl}}T_{\text{ctrl}}, (k_{\text{ctrl}}+1)T_{\text{ctrl}}]$ and $(u,d) \in L$, as

$$n_{u,d}(k_d+1) = f\left(n_{u,d}(k_d), g(k_{\text{ctrl}}), d(k_d)\right) \tag{9}$$

where $n_{u,d}(k_d)$ is the number of vehicles in link (u,d) at simulation time step k_d; $d(k_d)$ is the predicted disturbance (the traffic demand), which is the input traffic

flow rate for the network in the future; $g(k_{\text{ctrl}})$ is the future control input.

2) **Optimization problem**. Given the control interval T_{ctrl} and the prediction horizon N_{p}, the optimization problem of MPC with Total Time Spent (TTS) as objective function can be expressed as

$$\begin{aligned} \min_{\mathbf{g}(k_{\text{ctrl}})} J = & \sum_{k_d=NN_d k_{\text{ctrl}}+1}^{NN_d(k_{\text{ctrl}}+N_{\text{p}})} \sum_{(u,d)\in L} c_d \cdot n_{u,d}(k_d) \\ s.t. \quad & \text{Prediction model} \\ & \Phi(\mathbf{g}(k_{\text{ctrl}})) = 0 \quad \text{(cycle time constraints)} \\ & g_{\min} \le \mathbf{g}(k_{\text{ctrl}}) \le g_{\max} \end{aligned} \tag{10}$$

where $\mathbf{g}(k_{\text{ctrl}})$ represents the future control input sequence for control step k_{ctrl} (e.g. the green time splits), i.e.

$$\begin{aligned} \mathbf{g}(k_{\text{ctrl}}) = & [g^{\text{T}}(k_{\text{ctrl}}|k_{\text{ctrl}})\ g^{\text{T}}(k_{\text{ctrl}}+1|k_{\text{ctrl}}) \cdots \\ & g^{\text{T}}(k_{\text{ctrl}}+N_{\text{p}}-1|k_{\text{ctrl}})]^{\text{T}} \ , \end{aligned}$$

and the vector $g(k_{\text{ctrl}}+j|k_{\text{ctrl}})$ denotes the control input at the jth control step in the future from the current control time step k_{ctrl}. To decrease the on-line computational complexity, a control horizon N_{c} ($N_{\text{c}} < N_{\text{p}}$) can be defined, such that

$$\begin{aligned} g(k_{\text{ctrl}}+j|k_{\text{ctrl}}) = & g(k_{\text{ctrl}}+N_{\text{c}}-1|k_{\text{ctrl}}) \\ & \text{for } j = N_{\text{c}}, \cdots, N_{\text{p}}-1. \end{aligned}$$

3) **Rolling horizon**. Once the optimal control input $\mathbf{g}^*(k_{\text{ctrl}})$ is derived from the optimization, the first sample of the optimal result, $g^*(k_{\text{ctrl}} \mid k_{\text{ctrl}})$, is transferred to the process and implemented. When arriving to the next control step $k_{\text{ctrl}}+1$, the prediction model is fed with the real measured traffic states, the whole prediction horizon is shifted one step forward, and the optimization starts over again. This rolling horizon scheme closes the control loop, enables the system to get feedback from the real traffic network, and makes the MPC controller robust to uncertainty and disturbances.

Due to the nonlinear nature of the prediction model, the optimization problem of the MPC controller in (10) is a nonlinear non-convex optimization problem. Consequently, the MPC controller will become real-time infeasible when the scale of the controlled traffic network grows. Therefore, we will now reformulate this nonlinear non-convex optimization problem into an optimization problem that can be solved more efficiently.

IV. Reformulation of the Urban Traffic Model

The nonlinear non-convex optimization problem (10) can be reformulated into a mixed-integer linear optimization problem [29], which can be solved efficiently by existing MILP (Mixed-Integer Linear Programming) solvers [30]–[32]. The MILP solver is more efficient than the SQP solver for this particular optimization problem, and can find the global optimum rather than a local optimum.

A. Rules for Transformation

According to [33], consider the statement $f(x) \le 0$, where $f : \mathbb{R}^n \to \mathbb{R}$. Assume that $x \in \mathscr{X}$, where $\mathscr{X} \subset \mathbb{R}^n$ is a given bounded set, and define[2]

$$M = \max_{x\in\mathscr{X}} f(x), \quad m = \min_{x\in\mathscr{X}} f(x) \ . \tag{11}$$

Then, by introducing in $\delta \in \{0,1\}$, the following equivalence holds

$$\big([f(x) \le 0] \Leftrightarrow [\delta = 1]\big) \text{ iff } \begin{cases} f(x) \le M(1-\delta) \\ f(x) \ge \varepsilon + (m-\varepsilon)\delta \end{cases} \tag{12}$$

where ε is a small tolerance, typically the machine precision.

Moreover, $\delta f(x)$ can be replaced by the auxiliary real variable $z = \delta f(x)$ which satisfies $[\delta = 0] \Rightarrow [z = 0]$, $[\delta = 1] \Rightarrow [z = f(x)]$. Then $z = \delta f(x)$ is equivalent to

$$\begin{cases} z \le M\delta \\ z \ge m\delta \\ z \le f(x) - m(1-\delta) \\ z \ge f(x) - M(1-\delta) \end{cases} \tag{13}$$

B. Model Reformulation into Mixed-integer Linear Model

We now show how the model (2) can be reformulated as mixed-integer linear equations and inequalities using the equivalent reformulation rules above. Let

$$\begin{aligned} a &= \beta_{u,d,o}(k_d) \cdot \mu_{u,d} \cdot g_{u,d,o}(k_d)/c_d \\ b &= \big(q_{u,d,o}(k_d)/c_d\big) + \alpha^{\text{arriv}}_{u,d,o}(k_d) \\ c &= \beta_{u,d,o}(k_d)\big(C_{d,o} - n_{d,o}(k_d)\big)/c_d \\ d &= \min(a,b), \end{aligned} \tag{14}$$

then (2) becomes

$$\alpha^{\text{leave}}_{u,d,o}(k_d) = \min(a,b,c) = \min(d,c) \ . \tag{15}$$

Let

$$f_1 = b - a, \tag{16}$$

and define

$$\delta_1 = \begin{cases} 1 & \text{if } f_1 \le 0 \\ 0 & \text{if } f_1 > 0 \end{cases} \tag{17}$$

then we have

$$d = a + (b-a)\cdot\delta_1 = a + f_1 \cdot \delta_1 \ . \tag{18}$$

Similarly, let

$$f_2 = c - d, \tag{19}$$

and define

$$\delta_2 = \begin{cases} 1 & \text{if } f_2 \le 0 \\ 0 & \text{if } f_2 > 0 \end{cases} \tag{20}$$

then we have

$$\min(d,c) = d + (c-d)\cdot\delta_2 = d + f_2 \cdot \delta_2 \ . \tag{21}$$

Let

$$z_1 = f_1 \cdot \delta_1 \tag{22}$$

[2]In general, we could also take an upper or a lower bound instead of the maximum or the minimum.

$$z_2 = f_2 \cdot \delta_2 \tag{23}$$

and substitute (18) into (21), then (15) becomes linear, as

$$\alpha^{\text{leave}}_{u,d,o}(k_d) = a + z_1 + z_2 \ . \tag{24}$$

According to the equivalent transformation rules, (17) and (22) are equivalent to inequality constraints

$$\begin{aligned} & f_1 \le M_1(1-\delta_1) \\ & f_1 \ge \varepsilon + (m_1-\varepsilon)\delta_1 \\ & z_1 \le M_1\delta_1 \\ & z_1 \ge m_1\delta_1 \\ & z_1 \le f_1 - m_1(1-\delta_1) \\ & z_1 \ge f_1 - M_1(1-\delta_1) \ . \end{aligned} \tag{25}$$

Similarly, (20) and (23) are equivalent to inequality constraints

$$\begin{aligned} & f_2 \le M_2(1-\delta_2) \\ & f_2 \ge \varepsilon + (m_2-\varepsilon)\delta_2 \\ & z_2 \le M_2\delta_2 \\ & z_2 \ge m_2\delta_2 \\ & z_2 \le f_2 - m_2(1-\delta_2) \\ & z_2 \ge f_2 - M_2(1-\delta_2) \ . \end{aligned} \tag{26}$$

Here, M_1 and m_1 are the maximum value and the minimum value of f_1, and M_2 and m_2 are the maximum value and the minimum value of f_2, where $M_1 = C_{u,d}/c_d$, $m_1 = -\mu_{u,d}$, $M_2 = C_{d,o}/c_d$, and $m_2 = -\min(\mu_{u,d}, C_{u,d}/c_d)$.

Therefore, by introducing the additional auxiliary binary variables δ_1 and δ_2, and the auxiliary real variables f_1, f_2, z_1, and z_2, the original formula (2) in the urban traffic model is equivalently reformulated as linear equations (16), (19), and (24), and mixed-integer linear inequalities (25)-(26).

C. Reformulation of the Model Synchronization

Consider (8) for fixed i,u,d, and k_d. We will now show that this result in

$$\alpha^{\text{enter}}_{i,u,d}(k_d) = \mathscr{F}_{\text{in}}\big(\alpha^{\text{leave}}_{i,u,d}(k_u), \cdots, \alpha^{\text{leave}}_{i,u,d}(k_u+\ell)\big), \tag{27}$$

with ℓ an integer, and $\mathscr{F}_{\text{in}}$ a linear function.

In (8), $\alpha^{\text{leave,const}}_{i,u,d}(t)$ is a piecewise continuous function with intervals $\xi_{k_u}, \cdots, \xi_{k_u+\ell}$ and function values $\alpha^{\text{leave}}_{i,u,d}(k_u), \cdots, \alpha^{\text{leave}}_{i,u,d}(k_u+\ell)$. Hence, we have linear expression

$$\alpha^{\text{enter}}_{i,u,d}(k_d) = \frac{1}{c_d}\sum_{j=0}^{\ell} \xi_{k_u+j}\alpha^{\text{leave}}_{i,u,d}(k_u+j), \tag{28}$$

where ξ_x depends on c_d, c_u, and $\Delta c_{u,d}$. Once these variables are fixed, ξ_x is fixed.

The linear function $\mathscr{F}_{\text{in}}$ can be derived by the following approach. Given (8), we define

$$\begin{aligned} k_u^+ &= \text{floor}\left\{\frac{(k_d+1)\cdot c_d + \Delta c_{u,d}}{c_u}\right\}, \\ \theta_u^+ &= \text{rem}\left\{\frac{(k_d+1)\cdot c_d + \Delta c_{u,d}}{c_u}\right\}, \\ k_u^- &= \text{floor}\left\{\frac{k_d\cdot c_d + \Delta c_{u,d}}{c_u}\right\}, \\ \theta_u^- &= \text{rem}\left\{\frac{k_d\cdot c_d + \Delta c_{u,d}}{c_u}\right\}. \end{aligned} \tag{29}$$

where $k_u^+ \ge k_u^-$ and $0 \le \theta^+ < c_u$, $0 \le \theta^- < c_u$. Then, we obtain

$$\begin{aligned} \alpha^{\text{enter}}_{i,u,d}(k_d) &= \frac{1}{c_d}\int_{k_u^- c_u+\theta_u^-}^{k_u^+ c_u+\theta_u^+} \alpha^{\text{leave,cont}}_{i,u,d}(t)\,\mathrm{d}t \\ &= \frac{1}{c_d}\Bigg[\int_{k_u^- c_u+\theta_u^-}^{(k_u^-+1)c_u} \alpha^{\text{leave}}_{i,u,d}(k_u^-)\,\mathrm{d}t \\ &\quad + \sum_{i=1}^{k_u^+-k_u^--1}\int_{(k_u^-+i)c_u}^{((k_u^-+i+1)c_u} \alpha^{\text{leave}}_{i,u,d}(k_u^-+i)\,\mathrm{d}t \\ &\quad + \int_{k_u^+ c_u}^{k_u^+ c_u+\theta_u^+} \alpha^{\text{leave}}_{i,u,d}(k_u^+)\,\mathrm{d}t\Bigg] \\ &= \frac{1}{c_d}\Big[(c_u-\theta_u^-)\alpha^{\text{leave}}_{i,u,d}(k_u^-) + \\ &\quad c_u\sum_{i=1}^{k_u^+-k_u^--1}\alpha^{\text{leave}}_{i,u,d}(k_u^-+i) + \theta_u^+\alpha^{\text{leave}}_{i,u,d}(k_u^+)\Big] \ . \end{aligned}$$

Then, the synchronization function (8) can be rewritten into a linear equation of the form (27). Due to the definition of the control time interval, the synchronization formula will be the same in each control time interval. Taking the case in Fig. 2(b) for example, the synchronization functions within one control time interval are

$$\alpha^{\text{enter}}_{i,u,d}(k_d) = \alpha^{\text{leave}}_{i,u,d}(k_u), \tag{30}$$

$$\begin{aligned} \alpha^{\text{enter}}_{i,u,d}(k_d+1) = \frac{1}{c_d}\Big[&(c_u-\Delta c_{u,d}-c_d)\alpha^{\text{leave}}_{i,u,d}(k_u) \\ &+ (\Delta c_{u,d}+2c_d-c_u)\alpha^{\text{leave}}_{i,u,d}(k_u+1)\Big], \end{aligned} \tag{31}$$

$$\begin{aligned} \alpha^{\text{enter}}_{i,u,d}(k_d+2) = \frac{1}{c_d}\Big[&(2c_u-\Delta c_{u,d}-2c_d)\alpha^{\text{leave}}_{i,u,d}(k_u+1) \\ &+ \Delta c_{u,d}\alpha^{\text{leave}}_{i,u,d}(k_u+2)\Big], \end{aligned} \tag{32}$$

where $k_d = N_d k_{\text{ctrl}}$ and $k_u = N_c k_{\text{ctrl}}$. Therefore, the linear synchronization relationship can be pre-specified explicitly according to the given cycle times c_d and c_u of the corresponding intersections.

When the flow rate leaving link (u,d) is computed in the S model, the number of vehicles in downstream links $n_{d,o}(k_o)$ is used to calculate the number of vehicles that the downstream links can accept at most. The simulation time step counter of intersection o is k_o. If k_o is different from k_d, an output synchronization function is needed for synchronizing the original number of vehicles in the downstream link of link (u,d), $n^{\text{origin}}_{d,o}(k_o)$, from time step k_o to k_d, as

$$n_{d,o}(k_d) = \mathscr{F}_{\text{out}}\big(n^{\text{origin}}_{d,o}(k_o), \cdots, n^{\text{origin}}_{d,o}(k_o+\ell)\big), \tag{33}$$

which is also a linear expression that can be derived using the same rules as deriving the input synchronization function above.

D. Link Time Delay Assumption

Assumption 1: We assume that the time delay of the vehicles traveling from the beginning of the link to the end of the queues in the link is constant.

Then, having Assumption 1, (4) becomes linear as

$$\alpha^{\text{arriv}}_{u,d}(k_d) = (1-\gamma_{\text{const}})\cdot\alpha^{\text{enter}}_{u,d}(k_d-\tau_{\text{const}}) + \gamma_{\text{const}}\cdot\alpha^{\text{enter}}_{u,d}(k_d-\tau_{\text{const}}-1), \tag{34}$$

where τ_{const} and γ_{const} are constant values obtained by (5) with the queue length fixed. This queue length can be pre-calibrated for different traffic scenarios and environments according to the historical data, and stored in a data base.

With the reformulations above, the S model is reformulated into a mixed-integer linear model. Thereafter, an MILP method can be used to solve the optimization problem of the MPC controller based on the mixed-integer linear prediction model.

V. MILP-based MPC Controller

For intersection d, the control time interval and the simulation time interval satisfy $T_{\text{ctrl}} = NN_d c_d$. Then, for a given control time step k_{ctrl}, the corresponding simulation time steps are $k_d = NN_d k_{\text{ctrl}}, NN_d k_{\text{ctrl}}+1, \cdots, NN_d(k_{\text{ctrl}}+1)-1$.

After the model reformulation, the optimization problem of the MPC controller can be expressed as an MILP problem of the following form:

$$\begin{aligned} \min_{\mathbf{u}(k_{\text{ctrl}})} \; & J_{\text{TTS}} = \mathbf{c}^{\text{T}}\cdot\mathbf{u}(k_{\text{ctrl}}) \\ s.t. \quad & \mathbf{A}\mathbf{u}(k_{\text{ctrl}}) \le \mathbf{b} \\ & \mathbf{A}_{\text{eq}}\mathbf{u}(k_{\text{ctrl}}) = \mathbf{b}_{\text{eq}} \\ & \mathbf{u}_{\min} \le \mathbf{u}(k_{\text{ctrl}}) \le \mathbf{u}_{\max} \end{aligned} \tag{35}$$

for appropriately defined matrices $\mathbf{A}$, $\mathbf{A}_{\text{eq}}$, and vectors $\mathbf{c}$, $\mathbf{b}$, $\mathbf{b}_{\text{eq}}$, $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$, where $\mathbf{u}(k_{\text{ctrl}})$ contains all the optimization variables including the control inputs, the states, and the auxiliary variables for control time steps $k_{\text{ctrl}}, \cdots, k_{\text{ctrl}}+N_{\text{p}}-1$; all the inequality constraints and equality constraints are linear, and derived from the previous steps.

The vector of optimized variables at control time step k_{ctrl} in optimization problem (35) is

$$\begin{aligned} \mathbf{u}(k_{\text{ctrl}}) = [& u^{\text{T}}(k_{\text{ctrl}}|k_{\text{ctrl}})\ u^{\text{T}}(k_{\text{ctrl}}+1|k_{\text{ctrl}})\ \cdots \\ & u^{\text{T}}(k_{\text{ctrl}}+N_{\text{p}}-1|k_{\text{ctrl}})]^{\text{T}}\ , \end{aligned} \tag{36}$$

where $u(k_{\text{ctrl}}+j|k_{\text{ctrl}})$ at any control time step consists of control variables (i.e. green time splits), state variables, and auxiliary variables for all the nodes and links in the traffic network as:

$$\begin{aligned} u(\cdot) = [\quad & \overbrace{g^{\text{T}}(\cdot)}^{\text{Control variables}} \\ & \overbrace{q^{\text{T}}(\cdot)\ n^{\text{T}}(\cdot)\ n^{\text{T}}_{\text{downLink}}(\cdot)\ \alpha^{\text{T}}_{\text{leave}}(\cdot)\ \alpha^{\text{T}}_{\text{arriv}}(\cdot)\ \alpha^{\text{T}}_{\text{enter}}(\cdot)}^{\text{State variables}} \\ & \overbrace{\delta^{\text{T}}_1(\cdot)\ \delta^{\text{T}}_2(\cdot)\ f^{\text{T}}_1(\cdot)\ f^{\text{T}}_2(\cdot)\ z^{\text{T}}_1(\cdot)\ z^{\text{T}}_2(\cdot)}^{\text{Auxiliary variables}}]^{\text{T}}\ . \end{aligned} \tag{37}$$

where $(\cdot)$ stands for $(k_{\text{ctrl}}+j|k_{\text{ctrl}})$, $n^{\text{T}}_{\text{downLink}}(k_{\text{ctrl}})$ represents the vector of the state variables giving the number of vehicles in the downstream links. All the optimized variables are real values except for the binary variables $\delta_1(k_{\text{ctrl}})$ and $\delta_2(k_{\text{ctrl}})$. Supplied with initial traffic states and traffic demands of the network, the optimization problem can be solved at each control time step k_{ctrl} by the MILP solver. The optimal control inputs for the first control time step will be applied to the traffic network. Rolling one step ahead, a new MILP optimization problem will be built and solved, etc.

Several efficient branch-and-bound algorithms [30] are available for MILP problems. Moreover, there already exist several commercial and free solvers for MILP problems such as, e.g., CPLEX, Xpress-MP, GLPK, or lp_solve (see [31], [32] for an overview).

VI. S* Model-based MPC Controller via MILP

A. S* Model

For the S model described in Section II, the formula (2) computing the average flow rate leaving link (u,d) is the minimum of three terms. Each term gives the possible leaving flow rate under a traffic scenario. Under the saturated scenario, the average leaving flow rate depends on the saturated flow rate and the green time of the link; under the unsaturated scenario, the average flow rate is calculated according to the waiting and arriving flow rate at the intersection; under the over-saturated scenario, the average flow rate depends on the flow rate that the downstream link can accept. The traffic is always in the scenario that has the minimal average flow rate that could possible leave the link. As an urban traffic model, the S model is capable of describing all the situations that may happen in reality. However, when the S model is taken as a control model of the MPC controller, the third part of (2) can be removed from the S model to leave the over-saturated scenario out by adding extra constraints. Therefore, the S model can be rewritten into S* model by rephrasing (2) by

$$\begin{aligned} \alpha^{\text{leave}}_{u,d,o}(k_d) = \min\Big(& \beta_{u,d,o}(k_d)\cdot\mu_{u,d}\cdot g_{u,d,o}(k_d)/c_d, \\ & q_{u,d,o}(k_d)/c_d + \alpha^{\text{arriv}}_{u,d,o}(k_d)\Big)\ , \end{aligned} \tag{38}$$

and adding upper bound constraint $0 \le n_{u,d}(k_d) \le C_{u,d}$ to traffic state $n_{u,d}(k_d)$ (number of vehicles in a link) to make sure that the number of vehicles inside a link will not exceed its storage capacity $C_{u,d}$, i.e. no more vehicles can enter the link when it is already totally congested.

B. S* Model-based MPC Controller

An MPC controller can be established based on the S* model using the same method as shown in Section V. A similar MILP optimization problem as (35) can be built through reformulating the S* model into a mixed-integer linear model. But, for the new MILP optimization problem, the number of the auxiliary variables is reduced by half because of the reduction of the S* model. Although the S* model does not take the over-saturated scenario into consideration, it can still be guaranteed due to the constraints added. Instead of constraining the average traffic flow rates leaving links, the maximum number of vehicles that the downstream link can accept is then constrained by the upper bound. The traffic state $n(k)$, which is the number of vehicles in a link, is already an optimization variable of the MILP optimization problem. Hence, no extra effort is needed to add constraints to the traffic states $n(k)$ of all the links within the network

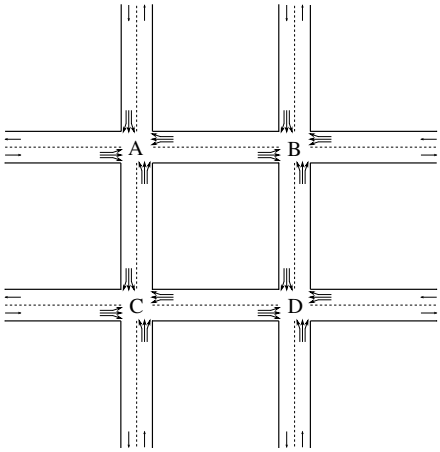


Fig. 3. The layout of an urban road network

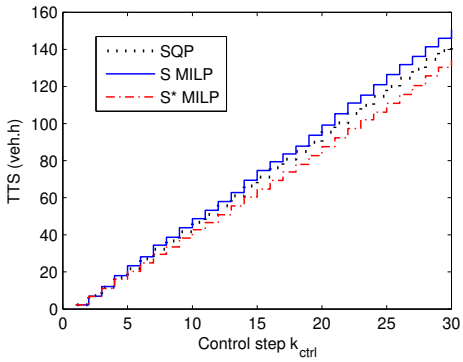


Fig. 4. TTS comparison of the approaches, SQP, S MILP, and S* MILP, for 500 veh/h

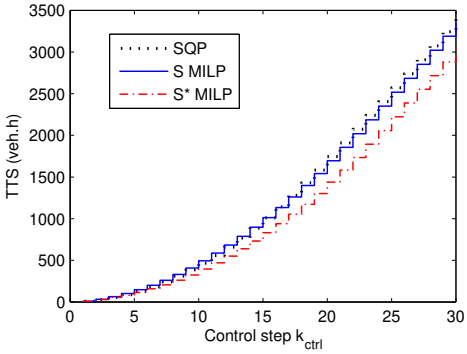


Fig. 5. TTS comparison of the approaches, SQP, S MILP, and S* MILP, for 2000 veh/h

at every simulation time step k. In fact, the key idea of this approach lies in simplifying the optimization problem by reducing one equation in the prediction model (the S model), and adding upper bounds to the optimized state variables $n(k)$ instead. As a result, the main complexity of the S* model-based optimization problem (the number of auxiliary integer variables introduced) is reduced by half. Meanwhile, even though the S* model-based MILP problem differs from the S model-based MILP problem, in most of the situations, the S* model-based MILP can keep a good control performance value compared with the S model-based MILP.

VII. Simulations

CORSIM is a microscopic traffic simulation software developed by FHWA [34], which can be used as a benchmark to design or test traffic control algorithms. We use CORSIM to simulate the real traffic environment, and design MPC controllers to decide control inputs for the traffic signals in CORSIM. The on-line optimization of the MPC controller is reformulated into different optimization problems, which are solved using different optimization methods, and then the control performance (TTS) of the MPC controllers are compared. Multi-start Sequential Quadratic Programming (SQP) is applied to solve the original S model-based nonlinear non-convex optimization problem. An MILP solver is used to solve the S model-based or S* model-based MILP problems obtained after reformulation according to Section IV.

As MILP solver, we use CPLEX, implemented through the `cplex` interface function of the Matlab Tomlab toolbox. For the SQP solver, we apply `fmincon` provided by the optimization toolbox of Matlab.

The urban traffic network investigated is a grid network including 4 intersections (see Fig. 3). The cycle times are 120 s for intersection A and D, and 60 s for intersection B and C. The cycle times are constant, and off-sets are 0 during the simulation (they can be further optimized in a higher control level). The control time interval is set to the least common multiple of all the cycle times in the network, i.e. T_{ctrl} is 120 s. The prediction horizon is 10 control intervals. The control simulations run for the same time period of 3600s for all the experiments. The length of the links are 1220 m, all the links have 3 lanes. The average vehicle length is 5 m, and the free-flow speed on the link is 50 km/h. Therefore, the storage capacity of each link in the network is 740 veh, and the constant time delay is set to be 87.8 s. The input traffic flow rates to the network are constant. The simulations are carried out under 3 scenarios, according to different values of the input traffic flow rates supplied to the network (input traffic demands), i.e. 500 veh/h, 2000 veh/h, and 3000 veh/h. The simulation results are compared for these different traffic scenarios. The cost function is TTS for the entire simulation. The number of initial points for the SQP algorithm is 5.

MPC controllers are built for the urban traffic network based on different optimization algorithms. The MILP approaches for the reformulated S model and the reformulated S* model are called respectively "S MILP" and "S* MILP" here. The control performance (TTS) of the controllers at every control step is extracted from CORSIM, and compared in Fig. 4 to Fig. 6 for all the Scenarios. In general, both S MILP and S* MILP have either better performance (lower TTS) than, or equal performance to the nonlinear optimization algorithm, SQP. The reason is that the optimization problem at hand is a nonlinear non-convex problem because of the nonlinearity of the S model, so that it may have multiple local optima. The SQP algorithm is only able to search for the local optimum, which in general results in a sub-optimal solution. A multi-start method can be applied to help select a better sub-optimal

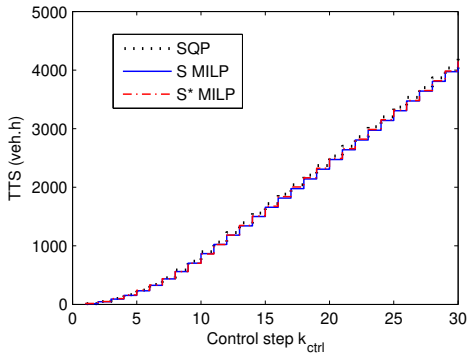


Fig. 6. TTS comparison of the approaches, SQP, S MILP, and S* MILP, for 3000 veh/h

solution. However, the multi-start procedure also results in more CPU time. On the contrary, an MILP problem can be solved efficiently by existing solvers that guarantee the global optimum.

We can see from Fig. 4 that the SQP algorithm has better performance (lower TTS) than S MILP, when the traffic flow demands are low. This is mainly caused by Assumption 1 made during the model reformulation. Recall that in order to turn the optimization problem into an MILP problem, Assumption 1 is made to linearize the original model. In the assumption, the time delay for vehicles running from the beginning of the link to the end of the queues in the link is considered to be constant. In the situation with high traffic demands, the number of leaving vehicles depends on the saturated flow rate of the link. In that case, the assumption almost does not have any influence on the results of MILP. However, in the situation with low traffic demands, the number of leaving vehicles from the link depends mainly on the number of waiting vehicles in the queues, which will be affected by the vehicles arriving from upstream after a certain time delay in the link. Therefore, the assumption causes a mismatch of the reformulated MILP problem from the original optimization problem. As a result, the MILP algorithm fails to achieve better results than the SQP algorithm, when the network is less crowded (low traffic demands). The reduced S* MILP is able to keep similar control performance as S MILP, in some situation even better.

When the traffic flow demands are high, and the traffic network is more crowded (saturated), the MILP approaches achieve better performance than the SQP approach, as Fig. 5 to Fig. 6 shows. The influence of Assumption 1, as in low demand scenarios, almost disappears. But, due to the high traffic demands and traffic density, there is also less space for the MILP approaches to improve the control performance, and hence, the TTS curves stay very close (see Fig. 6).

Furthermore, the output flow rates of the vehicles leaving the network from the north link of intersection A are also compared for the five scenarios. Fig. 7 to Fig. 9 illustrate the flow rates changing for each control time step during the whole simulations, when the input traffic flow rates supplied to the network are 500 veh/h, 2000 veh/h, and 3000 veh/h respectively. The average values of the output flow rates of the figures are also provided in Table I for the three algorithms, SQP, S MILP, and S* MILP, which illustrates that the S MILP approach obtains higher output average flow rates than the SQP approach when the input traffic flows are 2000 veh/h, but the other way round in the other scenarios. However, the figures demonstrate that curves of the network output flow rates are similar to each other for the three algorithms.

TABLE I
AVERAGE NETWORK OUTPUT FLOW RATES FOR DIFFERENT OPTIMIZATION ALGORITHMS IN ALL SCENARIOS (VEH/H)

Scenario	SQP	S MILP	S* MILP
500 veh/h	486	486	486
2000 veh/h	1031	1055	999
3000 veh/h	1079	945	986

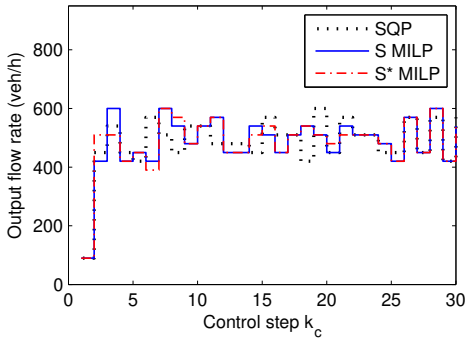


Fig. 7. Output flow rate comparison of the approaches for 500 veh/h

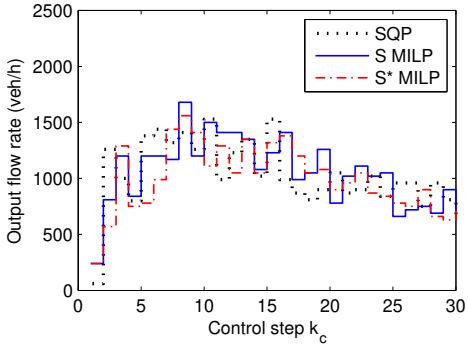


Fig. 8. Output flow rate comparison of the approaches for 2000 veh/h

In Table II, the computation time and the number of optimization variables are compared for different optimization algorithms, where "t_{avrg}" is the average optimization CPU time over all the control steps, and "t_{max}" is the maximum optimization CPU time. The SQP approach does not have boolean optimization variables. S* MILP has less optimization variables than S MILP, where the number of auxiliary variables is reduced by half because of the model adaptation. In general, the MILP problem with less boolean variables will be solved faster than the one with more boolean variables, due to the branch-and-search procedure of MILP solvers. But, this is not always true for the simulation results of S MILP and S* MILP. Nevertheless, S MILP and S* MILP problems can be both solved very fast by MILP solvers. The CPU

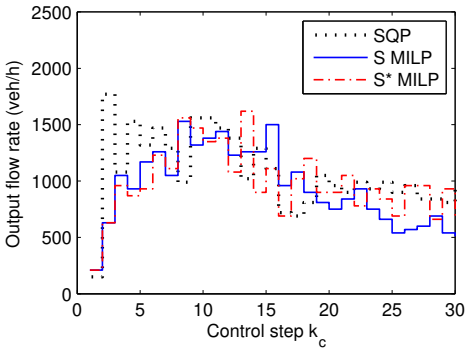


Fig. 9. Output flow rate comparison of the three approaches for 3000 veh/h

TABLE II
COMPARISON OF COMPUTATION TIMES AND THE NUMBER OF OPTIMIZATION VARIABLES FOR DIFFERENT OPTIMIZATION ALGORITHMS IN ALL SCENARIOS

Scenario	Algorithm	CPU time (s)		# variables	
		t_{avrg}	t_{max}	Real	Boolean
500 veh/h	SQP	461.4	601.7	120	-
	S MILP	0.8	2.9	6880	1440
	S* MILP	1.1	1.2	4480	720
2000 veh/h	SQP	453.4	552.5	120	-
	S MILP	1.2	2.3	6880	1440
	S* MILP	1.6	2.5	4480	720
3000 veh/h	SQP	452.4	526.4	120	-
	S MILP	1.1	2.6	6880	1440
	S* MILP	1.1	1.5	4480	720

times are reduced significantly from hundreds of seconds to a few seconds compared with the SQP solver. Therefore, by reformulating the original nonlinear non-convex optimization problem into an MILP problem, the MPC controller for urban traffic network becomes much more time efficient on-line. As a result, MILP methods are effective optimization approaches that can significantly reduce the on-line computational burden, and improve the real-life applicability of the MPC controllers for urban traffic networks.

As Table II shows, when the network demands are either very low or very high, the CPU time of the MILP algorithms is slightly lower than that for other network demands. The reason for this phenomenon lies on the principle of the MILP algorithm: The MILP algorithm uses a branch-and-search method to search for the optimal combination of all the integer variables, and a real-valued linear programming problem will be solved during each step of the search. When the network demands are very low or very high, the traffic will be in an unsaturated scenario or an over-saturated scenario. In an unsaturated scenario, the traffic flow rate for leaving a link will always depend on the vehicles waiting at the stop-line and arriving in the link. In an over-saturated scenario, the traffic flow rate for leaving a link will only depend on the available space in the downstream links. In both situations, most of the optimal integer variables of all the links in the network are already fixed, and need comparatively less time to find. But, if the network demands are neither very low nor very high, then all the three possible situations, as in (2), may happen randomly, thus more time will be needed to search for the optimal combination of the integer variables.

Although the MILP algorithms are very time efficient in the case study, in general, MILP problems are still NP-hard [35]. For larger urban traffic networks, if the size of the MILP problem becomes too large to be solved, other control structures can be adopted to avoid this problem. More specifically, a large-scale urban traffic network can be divided into several small subnetworks, which will be controlled and coordinated under a hierarchical control structure or a distributed control structure [18]–[22].

VIII. CONCLUSION

Model Predictive Control provides many advantages for controlling urban traffic networks. But it also has a high requirement for the computational efficiency of the on-line optimization. Due to the nonlinear non-convex nature of the optimization problem, the on-line computational complexity is a big challenge for the MPC controller. To solve this problem, in this paper, the nonlinear S model was reformulated into a mixed-integer linear model, which can be expressed by linear equalities and inequalities, through introducing auxiliary integer variables. The S model and the reduced S* model are both reformulated according to this method, and the original nonlinear non-convex optimization problem is written in the form of MILP problems based on the reformulated S model and S* model respectively. An efficient MILP solver can then be applied to solve the reformulated MILP optimization problems of MPC.

The simulation experiments show that the MILP approach is a breakthrough method to reduce the on-line computational complexity of the S model-based MPC controller, and also to increase the applicability of the MPC controller in real-life traffic networks.

In the future, multi-level control structures and algorithms will be further investigated for coordinating a large-scale urban traffic network from a higher level, taking this fast MILP-based MPC controller as the local subnetwork controller.

ACKNOWLEDGMENT

Research supported by a Chinese Scholarship Council (CSC) grant, the National Science Foundation of China (Grant No. 60674041, 60934007), the 7th framework European STREP project HD-MPC (contract number INFSO-ICT-223854), the BSIK projects “Next Generation Infrastructures (NGI)”, the Delft Research Center Next Generation Infrastructures, and the Transport Research Centre Delft.

REFERENCES

[1] D. Robertson and R. Bretherton, “Optimizing networks of traffic signals in real time - The SCOOT method,” *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 11–15, 1991.

[2] D. Bretherton, M. Bodger, N. Baber, and S. Controls, “SCOOT-the future [urban traffic control],” in *Proc. 12th IEEE International Conference on Road Transport Information and Control*, Apr. 2004, pp. 301–306.

[3] P. Lowrie, “The Sydney coordinated adaptive traffic system: principles, methodology, algorithms,” in *Proc. the International Conference on Road Traffic Signalling*, Mar. 1982, pp. 67–70.

[4] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, “Review of road traffic control strategies,” *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2043–2067, 2003.

[5] N. Gartner, "Simulation study of OPAC: A demand-responsive strategy for traffic signal control," *Transportation and Traffic Theory*, pp. 233–250, 1983.

[6] J. Farges, J. Henry, and J. Tufal, "The PRODYN real-time traffic algorithm," in *Proc. 4th IFAC Symposium of Transportation Systems*, Baden Baden, Germany, Jun. 1983, pp. 307–312.

[7] F. Boillot, J. Blosseville, J. Lesort, V. Motyka, M. Papageorgiou, and S. Sellam, "Optimal signal control of urban traffic networks," in *Proc. of Conference on Road Traffic Monitoring and Control*, Apr. 1992, pp. 75–79.

[8] S. Sen and K. Head, "Controlled optimization of phases at an intersection," *Transportation Science*, vol. 31, no. 1, pp. 5–17, 1997.

[9] C. Bielefeldt and F. Busch, "MOTION-a new on-line traffic signal network control system," in *Proc. the 7th International Conference on Road Traffic Monitoring and Control*, London, England, Apr. 1994, pp. 55–59.

[10] V. Mauro and C. Di Taranto, "Utopia," in *Proc. 2nd IFAC-IFIP-IFORS Symposium on Traffic Control and Transportation Systems*, 1989, pp. 575–597.

[11] C. Diakaki, M. Papageorgiou, and K. Aboudolas, "A multivariable regulator approach to traffic-responsive network-wide signal control," *Control Engineering Practice*, vol. 10, no. 2, pp. 183–195, 2002.

[12] K. Aboudolas, M. Papageorgiou, and E. Kosmatopoulos, "Store-and-forward based methods for the signal control problem in large-scale congested urban road networks," *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 2, pp. 163–174, 2009.

[13] J. Sanchez-Medina, M. Galan-Moreno, and E. Rubio-Royo, "Traffic signal optimization in "la almozara" district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 132 –141, Mar. 2010.

[14] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 630 –638, Sep. 2010.

[15] M. Dotoli, M. P. Fanti, and C. Meloni, "A signal timing plan formulation for urban traffic control," *Control Engineering Practice*, vol. 14, no. 11, pp. 1297–1311, 2006.

[16] M. van den Berg, A. Hegyi, B. De Schutter, and J. Hellendoorn, "Integrated traffic control for mixed urban and freeway networks: A model predictive control approach," *European Journal of Transport and Infrastructure Research*, vol. 7, no. 3, pp. 223–250, 2007.

[17] A. Hegyi, B. De Schutter, and J. Hellendoorn, "Optimal coordination of variable speed limits to suppress shock waves," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 1, pp. 102–112, Mar. 2005.

[18] N. H. Gartner, F. J. Pooran, and C. M. Andrews, "Implementation of the OPAC adaptive control strategy in a traffic signal network," in *Proc. of the 2001 IEEE International Intelligent Transportation Systems Conference*, Oakland (CA), USA, Aug. 2001, pp. 195–200.

[19] F. Boillot, S. Midenet, and J. C. Pierrelee, "The real-time urban traffic control system CRONOS: Algorithm and experiments," *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 1, pp. 18–38, 2006.

[20] P. Mirchandani and L. Head, "A real-time traffic signal control system: Architrcture, algorithm, and analysis," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 6, pp. 415–432, 2001.

[21] A. Di Febbraro, D. Giglio, and N. Sacco, "Urban traffic control structure based on hybrid petri nets," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 224–237, Dec. 2004.

[22] A. Kotsialos and M. Papageorgiou, "Efficiency and equity properties of freeway network-wide ramp metering with AMOC," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 6, pp. 401–420, 2004.

[23] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "Study on fast model predictive controllers for large urban traffic networks," in *Proc. the 12th International IEEE Conference on Intelligent Transportation Systems*, St. Louis, Missouri, USA, Oct. 2009, pp. 691–696.

[24] H. Kashani and G. Saridis, "Intelligent control for urban traffic systems," *Automatica*, vol. 19, no. 2, pp. 191–197, 1983.

[25] M. van den Berg, A. Hegyi, B. De Schutter, and J. Hellendoorn, "A macroscopic traffic flow model for integrated control of freeway and urban traffic networks," in *Proc. of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii USA, Dec. 2003, pp. 2774–2779.

[26] S. Lin and Y. Xi, "An efficient model for urban traffic network control," in *Proc. the 17th World Congress The International Federation of Automatic Control*, Seoul, Korea, Jul. 2008, pp. 14 066–14 071.

[27] S. Lin, B. De Schutter, Y. Xi, and J. Hellendoorn, "A simplified macroscopic urban traffic network model for model-based predictive control," in *Proc. 12th IFAC Symposium Control Transportation Systems*, Redondo Beach (CA), USA, Sep. 2009, pp. 286–291.

[28] E. Camacho and C. Bordons, *Model Predictive Control in the Process Industry*. Berlin, Germany: Springer-Verlag, 1995.

[29] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "Model predictive control for urban traffic networks via MILP," in *Proc. 2010 American Control Conference*, Baltimore (MD), USA, Jun. 2010.

[30] R. Fletcher and S. Leyffer, "Numerical experience with lower bounds for MIQP branch-and-bound," *SIAM Journal on Optimization*, vol. 8, no. 2, pp. 604–616, May 1998.

[31] A. Atamtürk and M. Savelsbergh, "Integer-programming software systems," *Annals of Operations Research*, vol. 140, no. 1, pp. 67–124, Nov. 2005.

[32] J. Linderoth and T. Ralphs, "Noncommercial software for mixed-integer linear programming," *Optimization Online*, Jan. 2005.

[33] D. Christiansen, *Electronics Engineers' Handbook*, 4th ed. New York: IEEE Press/McGraw Hill, 1997.

[34] FHWA, *Traffic software integrated system version 5.1 user's guide*, 2001.

[35] M. Garey and D. Johnson, *Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences*. WH Freeman and Company, San Francisco, Calif, 1979.