Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 54-2 (2021) 284–291

Parametrized Model Predictive Control Approaches for Urban Traffic Networks

Joost Jeschke* Bart De Schutter*

* *Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands (e-mails: joostjeschke@gmail.com, b.deschutter@tudelft.nl)*

Abstract: Model Predictive Control (MPC) has shown promising results in the control of urban traffic networks, but unfortunately it has one major drawback. The, often nonlinear, optimization that has to be performed at every control time step is computationally too complex to use MPC controllers for real-time implementations (i.e. when the online optimization is performed within the control time interval of the controlled network). This paper proposes an effective parametrized MPC approach to lower the computational complexity of the MPC controller. Two parametrized control laws are proposed that can be used in the parametrized MPC framework, one based on the prediction model of the MPC controllers, and another is constructed using Grammatical Evolution (GE). The performance and computational complexity of the parametrized MPC approach is compared to a conventional MPC controller by performing an extensive simulation-based case study. The simulation results show that for the given case study the parametrized MPC approach is real-time implementable while the performance decreases with less than 3% with respect to the conventional MPC controller.

Keywords: Grammatical Evolution, Parametrized Model Predictive Control, Urban Traffic Control, Parametrized Controller

1. INTRODUCTION

Due to growing populations and increasing economic activities, traffic congestion has become an urgent problem in urban areas. Traffic congestion results in negative social, economic, and environmental effects, and in longer travel times, which will be experienced as a loss of time and energy. Longer travel times will automatically result in the use of more fuel, which in its turn has a negative impact on the environment due to the emission of greenhouse gasses. The cost of traffic congestion for countries in the European Union (EU) is on average 1% of the country's GDP, resulting in a total cost of 110 billion euros a year in the EU alone (Christidis and Rivas (2012)). Due to the greenhouse gasses, traffic is responsible for approximately 50% of the NO_x emissions and 90% of the CO emissions (Nagurney (2000)). Also, with more congested urban areas, driver stress, aggression, and irritation increase (Hennessy and Wiesenthal (1999)), resulting in a higher chance of accidents happening.

Nowadays, traffic in urban areas is controlled by traffic lights to manage the flow of vehicles and to reduce the aforementioned negative effects. Model Predictive Control (MPC) is one of the control strategies on which a lot of research has been done and it has shown promising results in the control of urban traffic networks (Ye et al. (2019)). The main drawback of MPC is the online optimization that is performed at every control time step to calculate the control inputs. Due to the nonlinear behavior of traffic, nonlinear prediction models are used to predict future traffic states in the MPC framework (Lin and Xi (2008); Lin et al. (2012); Jamshidnejad et al. (2019)), resulting in a nonlinear and non-convex optimization problem that has to be solved online. This nonlinearity and non-convexity, and the need to control networks of intersections (resulting in a larger number of decision variables), results in a computationally complex optimization problem and makes MPC infeasible for real-time implementation in urban traffic networks.

Various approaches have been proposed to increase the real-time implementability of MPC for urban traffic networks, e.g. by reformulating the nonlinear optimization problem as a mixed integer linear programming problem (Lin et al. (2011)), by smoothening the nonlinear prediction model such that efficient gradient-based optimization algorithms can be used (Jamshidnejad et al. (2017)), or by solving the online optimization using a distributed (Ye et al. (2015)) or hierarchical (Ye et al. (2016)) approach such that small parts of the optimization problem can be efficiently solved in parallel.

Instead of dividing the optimization problem into smaller parts or using more efficient optimization methods, another method to lower the computational complexity is to reduce the number of decision variables in the optimization step by parametrizing the decision variables (Zegeye et al. (2012); Pippia et al. (2018)). In parametrized MPC, the original decision variables (i.e. the inputs of the system) are determined using a parametrized control law and the parameters of this control law become the new decision variables of the optimization problem. Parametrized MPC has shown promising results in control of freeway traffic networks (Zegeye et al. (2012)), substantially reducing the

10.1016/j.ifacol.2021.06.034

number of decision variables in the optimization problem with a limited decrease in performance.

In this paper, a parametrized MPC control approach for the control of urban traffic networks is proposed. Two parametrized control laws that can be used in the parametrized MPC control framework are designed. One is based on the prediction model of the MPC controllers and another is constructed using Grammatical Evolution (GE), a form of genetic programming. By using future states in the parametrized control laws, the number of decision variables can be reduced substantially with limited performance loss. This paper contributes to the state-of-the-art in three ways: by showing the possibilities of parametrized MPC, by proposing GE to construct a parametrized control law, and by improving the real-time implementability of MPC in urban traffic networks.

The remainder of this paper is organized as follows. In Section 2 the principles of conventional and parametrized MPC are discussed followed by a description of the urban traffic prediction model used in the parametrized MPC controllers in Section 3. In Section 4 the application of parametrized MPC in urban traffic networks is presented. In Section 5, GE will be introduced. The parametrized control law that will be used in the parametrized MPC controller is discussed in Section 6. The parametrized MPC controller will be compared to a conventional MPC controller in Section 7 for a case study network. Some final conclusions and suggestions for future work are given in Section 8.

2. (PARAMETRIZED) MODEL PREDICTIVE CONTROL

In this section, the principles of conventional and parametrized MPC will be shortly discussed.

2.1 Conventional MPC

In MPC (Rawlings and Mayne (2009)), an objective function and mathematical model are used to predict the future evolution of the system and to calculate an optimized control sequence. From this sequence, only the first set of inputs is implemented in the system, which is called the rolling horizon principle of MPC. The number of future states that are predicted corresponds to the prediction horizon N_p. A control horizon $N_\mathrm{c} < N_\mathrm{p}$ can be used to reduce the number of decision variables and the inputs are then held constant for the last $N_\mathrm{p} - N_\mathrm{c}$ steps.

MPC is a promising control method for urban traffic networks (Ye et al. (2019)) as 1) it can coordinate and control multiple control objectives at the same time, e.g., the total time spent by the vehicles in the network and the total emissions of the vehicles; 2) due to the rolling horizon approach, MPC can work with real-time feedback, allowing to quickly respond to changing traffic demands; 3) queue lengths and green times of the traffic lights can be constrained as MPC can take input and state constraints into account; and 4) the prediction model used to predict the future states can easily be updated or replaced by another model.

For large-scale systems or systems with fast dynamics, the (often nonlinear and non-convex) online optimization is the major drawback as it makes MPC hard to use for real-time implementation. As mentioned in the introduction, this paper will focus on reducing the computational complexity of the online optimization by reducing the number of decision variables in the optimization step. One of these methods is parametrized MPC (Zegeye et al. (2012)).

2.2 Parametrized MPC

The main idea of parametrized MPC is to reduce the number of decision variables by making the control inputs of the system dependent on a parametrized control law. The parameters of this control law become the new decision variables of the optimization problem and the control law is added to the constraints of the optimization problem (Zegeye et al. (2012)). The optimization problem that is solved at every control time step is:

$$\min_{\boldsymbol{\theta}} J(\tilde{\mathbf{x}}(k), \boldsymbol{\theta}) \tag{1}$$

$$\begin{aligned} \text{s.t.} \quad \mathbf{x}(k+j) &= f(\mathbf{x}(k+j-1), \mathbf{u}(k+j-1)), \\ \mathbf{u}(k+j-1) &= \mu(\mathbf{x}(k+j-1), \boldsymbol{\theta}), \\ g(\tilde{\mathbf{x}}(k), \tilde{\mathbf{u}}(k)) &\geq 0, \end{aligned}$$

for $j \in \{1, \ldots, N_\mathrm{p}\}$ and with

$$\begin{aligned} \tilde{\mathbf{x}}(k) &= \left[\mathbf{x}^\top(k+1), \mathbf{x}^\top(k+2), \ldots, \mathbf{x}^\top(k+N_\mathrm{p})\right]^\top, \\ \tilde{\mathbf{u}}(k) &= \left[\mathbf{u}^\top(k), \mathbf{u}^\top(k+1), \ldots, \mathbf{u}^\top(k+N_\mathrm{p}-1)\right]^\top, \end{aligned}$$

containing the future states and control inputs, respectively, $\boldsymbol{\theta}$ the parameters, $J(\cdot)$ and $f(\cdot)$ the cost function and prediction model of the MPC controller, respectively, and $\mu(\cdot, \boldsymbol{\theta})$ the parametrized control law. Note that there is no control horizon N_c as the states will be predicted up to time step N_p and the control inputs are a function of these states. For simplicity, the formulation of the parametrized control law only depends on the states at the current time step, but states at previous control time steps could be used as well. By introducing the parameters $\boldsymbol{\theta}$, the number of decision variables is reduced (if the number of parameters is less than the number of inputs), which then might result in faster optimization. The main challenge of parametrized MPC is finding a parametrized control law that results in faster optimization without losing too much performance.

Remark: In (1), $\boldsymbol{\theta}$ is assumed to be constant over the prediction horizon, but the parameters can also be chosen to vary at every time step $\boldsymbol{\theta}(k)$, or the idea of move-blocking MPC (Cagienard et al. (2007)) can be used for the parameters. In move-blocking MPC, the control inputs are held constant over several time steps to reduce the number of decision variables. Four different possibilities for the variation of the parameters in the parametrized control laws are proposed in Zegeye et al. (2012): (1) no variation: the parameters are constant over the control horizon; (2) time-dependent parameters: the parameters change every time step; varying parameters at every time step will generally result in better control, but also in higher computation times as there are more decision variables in the optimization; (3) move-blocking parametrized MPC, where the parameters are held constant for several time

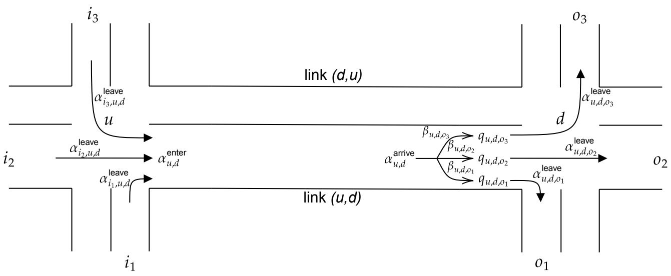
i_3
o_3
link (d,u)
$\alpha^{\text{leave}}_{i_3,u,d}$
u
d
$\alpha^{\text{leave}}_{u,d,o_3}$
β_{u,d,o_3}
q_{u,d,o_3}
$\alpha^{\text{leave}}_{i_2,u,d}$
i_2
$\alpha^{\text{enter}}_{u,d}$
$\alpha^{\text{arrive}}_{u,d}$
β_{u,d,o_2}
q_{u,d,o_2}
$\alpha^{\text{leave}}_{u,d,o_2}$
o_2
$\alpha^{\text{leave}}_{i_1,u,d}$
β_{u,d,o_1}
q_{u,d,o_1}
$\alpha^{\text{leave}}_{u,d,o_1}$
link (u,d)
i_1
o_1

Fig. 1. Link connecting two traffic-signal-controlled intersections based on Lin et al. (2012).

steps instead of the control inputs as in move-blocking MPC; (4) the principle of a control and prediction horizon for the parameters can be used; the parameters are then variable for some parameter control horizon and are held constant afterwards.

3. TRAFFIC PREDICTION MODEL

To illustrate the proposed approach we use the S-model (Lin et al. (2012); Jamshidnejad et al. (2019)) as prediction model throughout this paper as it provides a good balance between accuracy and complexity. Note however that the proposed approach is generic and that other traffic prediction models can also be used instead. The S-model is a macroscopic, nonlinear, and discrete-time model that uses the cycle time of a link's downstream intersection to update the states. Here, only the main equations of the model will be given that are needed to understand the remainder of the paper. For more details, we refer the reader to Lin et al. (2012); Jamshidnejad et al. (2019).

The S-model is defined by a set of nodes N, a set of links L, and a set of intersections $J \subset N$ (Fig. 1). A link $(u, d) \in L$ is defined by its upstream node $u \in N$ and downstream node $d \in N$, and has a set of input nodes $I_{u,d}$ and a set of output nodes $O_{u,d}$. The cycle times of the upstream and downstream node are given by c_u and c_d, respectively. For simplicity, in this paper the cycle times of all the intersections are chosen equal to the control interval of the network with time step counter k. The state variables of the model are the total number of vehicles $n_{u,d}(k)$ and the queue length $q_{u,d}(k)$ on each link (u, d). The states can take any non-negative value and are thus not restricted to integers. The queue lengths can be further divided in queue lengths for vehicles going to a specific output node $q_{u,d,o}(k)$ with $o \in O_{u,d}$. The number of vehicles and the queue lengths are updated every time step k by

$$n_{u,d}(k+1) = n_{u,d}(k) + \left(\alpha^{\text{enter}}_{u,d}(k) - \alpha^{\text{leave}}_{u,d}(k)\right) \cdot c_d, \quad (2)$$

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

$$q_{u,d}(k) = \sum_{o \in O_{u,d}} q_{u,d,o}(k), \quad (4)$$

where $\alpha^{\text{enter}}_{u,d}(k)$ and $\alpha^{\text{leave}}_{u,d}(k)$ are the average entering and leaving flow rates of link (u, d), $\alpha^{\text{arr}}_{u,d,o}(k)$ the average arriving flow rate at the tail of the queue of link (u, d) intending to move toward o, and $\alpha^{\text{leave}}_{u,d,o}(k)$ the average leaving flow rate of the sub-stream on link (u, d) moving towards o, in the time interval $[kc_d, (k+1)c_d)$. The leaving flow rates are nonlinear functions of the states, making the S-model nonlinear.

4. (PARAMETRIZED) MPC IN URBAN TRAFFIC NETWORKS

In this section, the objective function and the formulation of the optimization problems of the conventional and the parametrized MPC controllers are described.

4.1 Conventional MPC

At every time step, the MPC controller calculates the sub-optimal control inputs by solving the following optimization problem:

$$\begin{aligned} \min_{\tilde{\mathbf{g}}(k)} \quad & w_{\text{TTS}} \cdot \frac{J_{\text{TTS}}(k)}{\text{TTS}^{\text{n}}} + w_{\mathcal{D}} \frac{\mathcal{D}(\tilde{\mathbf{g}}(k))}{\mathcal{D}^{\text{n}}} + w_{\text{f}} \frac{J_{\text{final}}(k+N_{\text{p}})}{J^{\text{n}}_{\text{final}}} \\ \text{s.t.} \quad & \mathbf{x}_{u,d}(k+j) = f(\mathbf{x}_{u,d}(k+j-1), \mathbf{g}_d(k+j-1)), \quad (5) \\ & \mathcal{U}\left(\mathbf{g}_d(k+j)\right) = 0, \quad \text{for } j = 1, \ldots, N_{\text{p}}, \ \forall d \in J \\ & \mathbf{g}_{\min} \leq \tilde{\mathbf{g}}(k) \leq \mathbf{g}_{\max} \end{aligned}$$

where TTS stands for the total time spent by the vehicles in the links of the network during the prediction period, $\tilde{\mathbf{g}}(\mathbf{k}) = \left[\mathbf{g}^\top(k), \mathbf{g}^\top(k+1), \ldots, \mathbf{g}^\top(k+N_{\text{p}}-1)\right]^\top$, in which $\mathbf{g}(k)$ is a column vector with the phase times $\mathbf{g}_d(k)$ of all the intersections $d \in J$ at time step k, $\mathbf{x}_{u,d}(k)$ a vector with the number of vehicles and queue length of link (u, d) (i.e. $[n_{u,d}(k), q_{u,d}(k)]^\top$), $\mathbf{g}_{\min}$ and $\mathbf{g}_{\max}$ vectors of appropriate size with the minimum and maximum green times of the traffic lights, respectively, w_{TTS}, $w_{\mathcal{D}}$, and w_{f} the weights that describe the importance of the different control objectives, TTS^{n}, $\mathcal{D}^{\text{n}}$, and $J^{\text{n}}_{\text{final}}$ are nominal values of the different control objectives, and $\mathcal{U}(\mathbf{g}_d(k+j)) = 0$ describes the constraint on the phase times of an intersection, i.e.

$$\mathcal{U}(\mathbf{g}_d(k+j)) = c_d - y_d - \sum_{i=1}^{N^{\text{ph}}_d} g_{d,i}(k+j) \quad (6)$$

where y_d and N^{ph}_d are the total yellow time and the number of phases at intersection d, respectively, and $g_{d,i}$ the green time of the i-th phase at intersection d. The different parts of the objective function in (5) describe the total time spent, a cost on the control inputs, and a terminal cost to take the vehicles at the end of the horizon into account, respectively. We have

$$J_{\text{TTS}}(k) = \sum_{(u,d) \in L} \sum_{j=1}^{N_{\text{p}}} c_d \cdot n_{u,d}(k+j) \quad (7)$$

$$\begin{aligned} \mathcal{D}(\tilde{\mathbf{g}}(k)) = \Bigg\| \Big[& (\tilde{\mathbf{g}}(k) - \tilde{\mathbf{g}}(k-1))^\top, (\tilde{\mathbf{g}}(k+1) - \tilde{\mathbf{g}}(k))^\top, \\ & \ldots, \ (\tilde{\mathbf{g}}(k+N_{\text{c}}) - \tilde{\mathbf{g}}(k+N_{\text{c}}-1))^\top \Big]^\top \Bigg\|_2 \end{aligned} \quad (8)$$

$$J_{\text{final}}(k+N_{\text{p}}) = \left\| \left[\mathbf{n}(k+N_{\text{p}}), \mathbf{q}(k+N_{\text{p}}) \right]^\top \right\|_2 \quad (9)$$

4.2 Parametrized MPC

In parametrized MPC, the original decision variables are replaced by the parameters in (5)

$$\min_{\boldsymbol{\theta}} \; w_{\mathrm{TTS}} \cdot \frac{J_{\mathrm{TTS}}(k)}{\mathrm{TTS}^{\mathrm{n}}} + w_{\mathcal{D}} \frac{\mathcal{D}(\tilde{\mathbf{g}}(k))}{\mathcal{D}^{\mathrm{n}}} + w_{\mathrm{f}} \frac{J_{\mathrm{final}}(k+N_{\mathrm{p}})}{J^{\mathrm{n}}_{\mathrm{final}}}), \tag{10}$$

and the parametrized control law is added to the constraints in (5)

$$\mathbf{g}_d(k+j) = \mu(\mathbf{x}(k+j), \boldsymbol{\theta}) \quad \text{for } j = 0, \ldots, N_{\mathrm{p}} - 1, \tag{11}$$

where $\mu(\cdot, \boldsymbol{\theta})$ is the parametrized control law.

5. GRAMMATICAL EVOLUTION

GE is a form of genetic programming (Koza (1994)) that produces programs based on a user-defined context-free grammar (O'Neill and Ryan (2001)). Context-free grammars have a recursive notation and describe how programs or functions can be constructed from a list of variables and functions (Hopcroft et al. (2006)). A context-free grammar consists of four basic components: a finite set of terminals, a finite set of non-terminals, a start symbol, and a finite set of production rules. The production rules represent the recursive behavior of the context-free grammar.

An evolutionary algorithm is used to evolve the programs constructed with the context-free grammars (Nicolau and Agapitos (2018)). These programs can be complex programming languages, or more simplistic curve fitting models or symbolic regressions (Poli et al. (2008)). In our work, the programs are parametrized control laws. An initial population of programs is generated and from there, new populations are generated using mutation and crossover operations. The individual programs in a population have an underlying chromosome, which is an integer string that is mapped to the actual program using the grammar. Like other evolutionary algorithms, the programs are used to evaluate the performance of the individuals and the corresponding chromosome is used to evolve the program between consecutive generations.

Later on in this paper, GE will be used to construct a parametrized control laws that checks whether certain traffic conditions hold in the network and assigns an appropriate phase time accordingly. As we use an user-defined grammar, we can place the parameters in the conditional statements of the controller, which has huge benefits for the training time of the algorithm as will we show in the next section.

6. PARAMETRIZED CONTROL LAWS

In our (parametrized) MPC approach, the sum of the phase times at an intersection should add up to the available green time of that intersection (i.e. the cycle time minus the yellow times). This results in an equality constraint in the optimization problem that can be hard to satisfy with a parametrized control law. The parametrized control laws should be designed to satisfy this constraint at all times. Here, two parametrized control laws are proposed: one using the relative queue lengths and arriving flow rates, and one trained using GE.

6.1 Using the relative queue lengths

The leaving flow rate of a lane in the S-model depends on the traffic conditions on the lane. The leaving flow rate is, among others, a function of the arriving flow rate and the queue length. These relations are used in the design of the first parametrized control law. As the sum of the phase times should add up to the cycle time of the intersection, the available green time is divided over the phases based on the number of vehicles in the queues and the arriving flow rates with respect to the other phases.

Let us denote $q^{\mathrm{ph}}_{d,j}(k)$ as the mean queue length of the lanes that have right-of-way (i.e. a green light) in the j-th phase at intersection d at control time step k, and $\overline{q}^{\mathrm{ph}}_d(k)$ as the mean of $\mathbf{q}^{\mathrm{ph}}_d(k)$, which contains the mean queue lengths $q^{\mathrm{ph}}_{d,j}(k)$ of the phases at intersection d. In the same way, $\alpha^{\mathrm{ph,arr}}_{d,j}(k)$, $\boldsymbol{\alpha}^{\mathrm{ph,arr}}_d(k)$, and $\overline{\alpha}^{\mathrm{ph,arr}}_d(k)$ are defined for the arriving flow rates. The green time $g_{d,j}(k)$ of phase j at intersection d is calculated by [1]

$$g_{d,j} = \overline{g}_d + \frac{q^{\mathrm{ph}}_{d,j} - \overline{q}^{\mathrm{ph}}_d}{\sum_{i=1}^{N^{\mathrm{ph}}_d} q^{\mathrm{ph}}_{d,i} + \kappa_q} \cdot \theta_{d,1} + \frac{\alpha^{\mathrm{ph,arr}}_{d,j} - \overline{\alpha}^{\mathrm{ph,arr}}_d}{\sum_{i=1}^{N^{\mathrm{ph}}_d} \alpha^{\mathrm{ph,arr}}_{d,i} + \kappa_\alpha} \cdot \theta_{d,2} \tag{12}$$

where $\overline{g}_d$ is the mean green time during one cycle at intersection d (i.e. the cycle time minus the yellow time, divided by the number of phases), $\theta_{d,1}$ and $\theta_{d,2}$ are the parameters to be optimized, N^{ph}_d is the number of phases at intersection d, and κ_q and κ_α small positive values to prevent division by zero.

In the formulation of the control law in (12), the sum of both the part depending on the queue lengths and the part depending on the arriving flow rate add up to zero. Therefore, the equality constraint that the phase times should add up to the cycle time is always satisfied. To some of the phases some green time is added to the mean green time, while from other phases this green time is subtracted.

6.2 Using Grammatical Evolution

The second control law is constructed using GE. The program that is constructed with the grammar used (Figure 2) does not create a single expression, but creates a program consisting of multiple if/else-statements in which certain traffic conditions on the lanes are checked. The available green time is divided over the phases based on these conditions. In the conditions, it is checked whether (1) the relative queue of the phases, $q^{\mathrm{ph,rel}}_{d,j}$, which is the coefficient in front of $\theta_{d,1}$ in (12), is positive or negative as that tells something about the degree of congestion on the lanes compared to the other lanes; (2) all the vehicles that are initially in the queue, i.e. at the beginning of a cycle, can leave during this cycle if the phase times would be equal to the mean green time of the cycle; (3) the maximum queue length of the phases is higher than a certain threshold $\theta_{d,2}$.

The output of the grammar in Figure 2, $y_{d,j}$, is a score (which describes the importance of a certain phase with respect to the other phases) for the j-th phase of intersection d. The variables that are used in the grammar are $\mathbf{q}^{\mathrm{ph}}$, $\mathbf{q}^{\mathrm{ph,rel}}$, and $\mathbf{q}^{\mathrm{ph,max}}$, which are the mean, relative, and maximum queue length of phase j, respectively, the mean

[1] For the sake of simplicity of notation, the time step indicator k is left out in the equation.

```
⟨deff⟩    ::= $y_{d,j} = \bar{g}_d$
              if $q_{d,j}^{\mathrm{ph}}$ >=:
                  if $q_{d,j}^{\mathrm{ph,rel}}$ >= 0:
                      ⟨code⟩
                  else:
                      ⟨code⟩
              else:
                  if $q_{d,j}^{\mathrm{ph,rel}}$ >= 0:
                      ⟨code⟩
                  else:
                      ⟨code⟩
⟨code⟩    ::= ⟨if⟩ | ⟨stmt⟩
⟨if⟩      ::= if $q_{d,j}^{\mathrm{ph,max}}$ ⟨relop⟩ : ⟨if-opt⟩
⟨if-opt⟩  ::= ⟨stmt⟩ | ⟨stmt⟩ else: ⟨stmt⟩
⟨relop⟩   ::= >= | <= | > | <
⟨numif⟩   ::= 1 | ... | 4
⟨stmt⟩    ::= $y_{d,j} := y_{d,j}$ ⟨op⟩ ⟨expr⟩
⟨expr⟩    ::= ⟨expr⟩ ⟨op⟩ ⟨expr⟩ | ⟨number⟩ | ⟨var⟩
⟨op⟩      ::= + | - | * | /
⟨var⟩     ::= $q_{d,j}^{\mathrm{ph}}$ | $q_{d,j}^{\mathrm{ph,max}}$ | $q_{d,j}^{\mathrm{ph,rel}}$
⟨number⟩  ::= ⟨num⟩.⟨num⟩ | ⟨num⟩ | ⟨num⟩⟨num⟩
⟨num⟩     ::= 1 | ... | 9
```

Fig. 2. The user-defined grammar in Backus-Naur form to construct a parametrized control law for phase j of intersection d.

green time $\overline{g}_d$ at intersection d, and the saturated flow rate $\mu_{u,d}$. The phase times are calculated using the scores of the phases:

$$g_{d,j} = \frac{y_{d,j}}{\sum_{i=1}^{N_d^{\mathrm{ph}}} y_{d,i}} \cdot g_{d,\mathrm{tot}}. \tag{13}$$

In the above equation, a portion of the available green time is assigned to each phase, ensuring that the constraint on the cycle time always holds.

An extensive data set is collected for training the algorithm. The data set consists of sets of phase times per intersection, i.e. for an intersection with four phases, one data point consists of four phase times at a certain time step as the dependent variable, and the mean, relative, and maximum queue length of the phases at the same time step as the explanatory variables. The data is generated by simulating multiple scenarios that are controlled by the conventional MPC controller defined in Section 7.3. For training purposes, it is assumed that every generated data point is independent of the other data points, i.e. traffic at other intersections did not contribute to the phase times of the intersection considered in the data point.

In the optimization problem of the parametrized MPC controller, there are two parameters per intersection to calculate the phase times of that intersection. The parameters should therefore be optimized per data point as one data point represents the phase times of one intersection at one time step. Thus, during training of the control law with GE, the optimal values for $\theta_{d,1}$ and $\theta_{d,2}$ must be found for every data point. The parameter $\theta_{d,1}$ is included in the first if-statement of the <deff> production rule (see Figure 2) and represents a mean queue length. The second parameter is included in the conditional statement of the <if> production rule and represents a maximum queue length. As the parameters are in the if-statements, we can evaluate the constructed control law for all combinations of the mean queue length and maximum queue length in the explanatory variable to find the performance of the

```
$y_{d,j} = \bar{g}_d$
if $q_{d,j}^{\mathrm{ph}}$ >=:
    if $q_{d,j}^{\mathrm{ph,rel}}$ >= 0:
        if $q_{d,j}^{\mathrm{ph,max}}$ >:
            $y_{d,j} := y_{d,j} + 4.7$
        else:
            $y_{d,j} := y_{d,j} - 7.7$
    else:
        if $q_{d,j}^{\mathrm{ph,max}}$ <=:
            $y_{d,j} := y_{d,j} + 0.7$
        else:
            $y_{d,j} := y_{d,j} \times 0.5$
else:
    if $q_{d,j}^{\mathrm{ph,rel}}$ >= 0:
        if $q_{d,j}^{\mathrm{ph,max}}$ <=:
            $y_{d,j} := y_{d,j} - 0.9$
        else:
            $y_{d,j} := y_{d,j}/1.7$
    else:
        if $q_{d,j}^{\mathrm{ph,max}}$ >:
            $y_{d,j} := y_{d,j} + 7.3$
        else:
            $y_{d,j} := y_{d,j} - 7.3$
```

Fig. 3. The final parametrized control law constructed using GE.

program on a single data point. The performance over all the data points is the final performance of a program and the performance on which the control laws are compared.

7. CASE STUDY

To compare the parametrized MPC controllers to a conventional MPC controller, a simulation-based case study is performed. In the case study, three traffic scenarios are considered and simulated in the traffic simulator SUMO (Krajzewicz et al. (2002)). The interface TraCI (Wegener et al. (2008)) is used to communicate between SUMO and Matlab. The focus will be on the system performance, which is measured by the Total Time Spent (TTS) by the vehicles in the network, and the computational complexity. All the simulations are performed on an HP ZBook Studio G4 (containing an Intel Core i7 processor with a clocking speed of 2.8GHz and an 8GB RAM) using Matlab R2018b.

7.1 Setup

The urban traffic network that is used for the simulations is shown in Fig. 4. The model consists of three four-way intersections and one three-way intersection, seven origins, and 22 links, which all have three lanes, one for each turning direction. The length and maximum speed of a vehicle are 7 [m] and 12 [m/s], respectively. The lane lengths are 500 [m] except for links 11 and 22, which have a length of 700 [m], and links 5 and 16, which have a length of 540 [m]. The cycle time of each intersection is equal to 60 seconds and the model parameters of the S-model are identified offline and are given in Table 1. The traffic flow profiles are generated with SUMO's built-in route generator, which generates routes based on flow profiles and turning rates. The three scenarios lead to different degrees of congestion in the network and are referred to as the high-demand, two-peaks, and alternating-demand scenario (Fig. 5). The duration of the three scenarios is 3600 seconds (one hour) and they are initialized from an empty network followed by a traffic flow of 0.08 [veh/s]

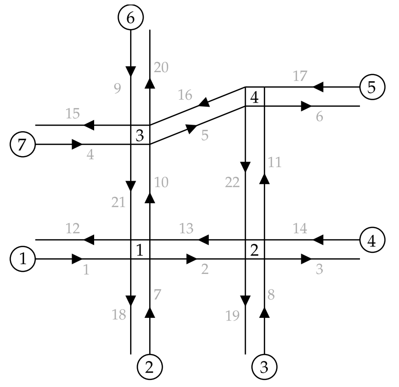

Fig. 4. Urban traffic network used for the case study.

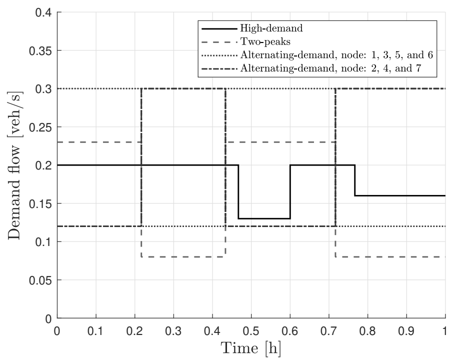

Fig. 5. The demand flow profiles of the different profiles.

from all origins for 5 minutes. The prediction horizon of the controllers is chosen to 5, such that a vehicle that enters the network is able to leave the network during the prediction window. The control horizon of the conventional MPC controller is chosen to be equal to the prediction horizon.

The cost functions from (5) and (10) are used for the conventional MPC and parametrized MPC controllers, respectively, with $w_{\mathcal{D}} = 1$, $\text{TTS}^{\text{n}} = 10^5$ [s], $\mathcal{D}^{\text{n}} = \|\mathbf{g}_{\max} - \mathbf{g}_{\min}\|_2$, and $J^{\text{n}}_{\text{final}} = 175$ [veh]. These nominal values are derived from their values during one prediction window, e.g. the total time spent during N_{p} time steps.

7.2 Performance measures

The computational complexity is measured by the processor time per control step. The mean computation time over all the optimization steps is considered to compare the computational complexity of the different controllers and the maximum computation time over all the optimization steps is used to see whether a controller could be used for real-time implementation. The relative change of the TTS and of the *mean* computation time (CT) with respect to the conventional MPC controller will be used as performance measures, e.g.

$$\text{TTS}^{\text{rel}} = \frac{\text{TTS}_{\text{PMPC}} - \text{TTS}_{\text{MPC}}}{\text{TTS}_{\text{MPC}}} \cdot 100\%,$$

for the TTS. In the same way, CT^{rel} is defined. The phase times calculated with the parametrized control laws are rounded before calculating the function values to make a fair comparison with the conventional MPC controller, which is restricted to integer values as SUMO uses integer values (i.e. seconds) to switch the traffic lights.

7.3 Controllers

The performance of the parametrized MPC controllers is compared to that of a conventional MPC controller and a fixed-time controller (from which the phase times are equal to the mean phase time of the intersections, i.e. the available green time divided by the number of phases). The optimization of the parametrized MPC controllers and conventional MPC controller will be solved using genetic algorithms as these algorithms outperformed other algorithms, such as SQP and simulated annealing, in preliminary experiments not reported here. Note that in general the solutions of the optimization problems are a good approximation of the global minimum, resulting in sub-optimal performance of the parametrized MPC controller. Here, the settings of the optimization problems of the (parametrized) MPC controllers are discussed. For all the controllers the following holds: if the phase times over the whole prediction horizon have not changed for 10 consecutive generations, the genetic algorithm is terminated.

Conventional MPC controller The linear equality constraint in (6) can be used to eliminate and substitute one of the phase times into the objective function. By using the equality constraint to eliminate variables, the number of decision variables is reduced from 75 to 55 for the traffic network used in the case study. The two-point crossover and adaptive feasible mutation operators in Matlab are slightly modified such that only feasible children are produced. The feasible two-point crossover makes sure that the chromosomes are only cut after the phase times of an intersection such that the inequality constraint on the phase times always holds. For the same reason, the feasible mutation algorithm mutates the values of one intersection from which the sum of the mutations adds up to zero to ensure that the inequality constraint always holds. Using these feasible operators instead of the standard operators in Matlab increased the system performance with more than 20% for the same computation time. Based on the results of some tuning experiments, the population size is set to 200 and the maximum optimization time per control step is set to 36 minutes.

Parametrized MPC - relative queue lengths The number of parameters in the control law in (12) is two per intersection, resulting in a total of 8 decision variables in the optimization problem as the parameters are held constant over the control horizon. Due to the formulation of the control law, the equality constraint (6) on the sum of the phase times is always satisfied and can be left out of the optimization problem. As the control law is linear in the parameters, the lower and upper bounds on the phase times can easily be rewritten as constraints on the parameters. There are no upper and lower bounds on the parameters.

The population size of the genetic algorithm is now 100 and the standard scattered crossover and Gaussian mutation operators in Matlab are used. The parameters are initialized in $-40 \leq \theta_{d,1}, \theta_{d,2} \leq 40$ for all intersections $d \in J$, to prevent the production of invalid individuals.

Table 1. S-model Parameters. $i \in \{1,3,4,6,7,8,9,12,14,15,17,18,19,20\}$, $j \in \{2,13\}$, $k \in \{5,16\}$, $l \in \{10,21\}$, and $m \in \{11,22\}$.

Parameter	μ [veh\s]	v^{free} [m\s]	v^{idle} [m\s]	a^{dec} [m/s^2]	l_i^{edge} [m]	l_j^{edge} [m]	l_k^{edge} [m]	l_l^{edge} [m]	l_m^{edge} [m]	l^{veh} [m]
Value	0.8436	12.8933	0.4773	-1.7697	492.465	482.941	554.755	457.924	760.871	7.9013

Parametrized MPC - Grammatical Evolution In the training process of the GE-based control law, all possible combinations of $\theta_{d,1}$ and $\theta_{d,2}$ were considered to find the performance of an individual (see Section 6.2). However, in the online optimization problem of the parametrized MPC controller, future states will be predicted which will differ from the initial queue lengths and maximum queue lengths. Therefore, the parameters cannot simply be the values of the queue lengths but will be continuous variables in the range $[0, 30]$ as queue lengths longer than 30 were not observed.

The population size is now set to 300 to search a range of the minima and from there the other regions are searched using crossover and mutation. With smaller population sizes, the algorithm might converge to a non-optimal minimum without discovering the more optimal regions of the search space. Both parameters will be initialized around the current queue length and maximum queue length, i.e. $\theta_{d,1} \in \left[\max(0, \min\left(\mathbf{q}_d^{\text{ph}}\right) - 10), \min(\max\left(\mathbf{q}_d^{\text{ph}}\right) + 10, 30)\right]$, for $d \in J$, and $\theta_{d,2}$ is initialized using the same bounds but with $\mathbf{q}_d^{\text{ph}}$ replaced by $\mathbf{q}_d^{\text{ph,max}}$. The standard crossover and Gaussian mutation operators in Matlab are used.

7.4 Results

In Table 2 the system performance and computational complexity of the different controllers are shown for the different demand scenarios. The parametrized MPC controllers reduce the computational complexity of the online optimization with more than 90% for all the demand scenarios with respect to the conventional MPC controller, while the system performance increases only up to 2.2% for the relative-queue-length-based parametrized MPC controller and 15% for the parametrized MPC controller using the GE-based control law. The differences in performance of the parametrized MPC controllers can be explained by the size of their search spaces. Although the search space is hard to define, one can imagine that the relative-queue-length-based parametrized control law has a larger search space than the GE-based parametrized control law as the latter is restricted to assign the scores in the different regions. Although the computation time of the online optimization is reduced considerably, the parametrized MPC controllers cannot be used for real-time implementation as the maximum computation time is higher than the control time interval of 60 seconds of the network.

As we strive for real-time implementability of the parametrized MPC controllers, all the simulations are performed again with a maximum computation time of 60 seconds for all the (parametrized) MPC controllers. In Table 3, the system performance of the conventional MPC controller with time constraints of 36 minutes and 60 seconds, and the two parametrized MPC controllers, are shown. What stands out is the enormous decrease in system performance ($\text{TTS}^{\text{rel}} = 25.2\%$) of the real-time implementable conventional MPC controller. This emphasizes the need for methods that improve real-time implementability of MPC controllers. The system performance of both parametrized MPC controllers remains approximately the same, from which we can assume that the controllers already converge to a good approximation of the global minimum in the first iterations of the algorithm. Some performances even improved which could be due to inaccuracies in the macroscopic prediction model.

8. CONCLUSIONS

In this paper, a parametrized MPC approach for control of urban traffic networks has been proposed. One of the parametrized control laws uses the relative queue lengths and arriving flow rates and the other is constructed using Grammatical Evolution (GE). The parameters are inserted in the conditional statements in the grammar of the GE framework, resulting in efficient training of the parametrized control law. To demonstrate the effectiveness of parametrized MPC, a simulation-based case study has been performed in which the system performance and computational complexity of the parametrized MPC controllers have been compared to that of a conventional MPC controller. For the urban traffic network used throughout this case study and using parametrized MPC, the number of decision variables decreased from 55 for the optimization problems of the conventional MPC controller to 8 for the optimization problems of the parametrized MPC controllers. This decreased the computational complexity of the relative-queue-length-based parametrized MPC controller with 97% while the system performance deteriorated with less than 3% on all three the demand profiles. Hence, parametrized MPC is a promising control strategy to make MPC real-time implementable.

In future work, more complex case studies should be considered to show the effectiveness of parametrized MPC on larger networks. Also, we could make the grammar of the GE framework more rich by using a set of predefined functions that are simply evaluated in every statement instead of using the arithmetic operations, which might increase the performance of the control law. During evolution, these predefined functions are then interchanged over the different regions using crossover and mutation. These predefined functions could, e.g., be the other proposed parametrized control law or pre-trained artificial neural networks for different traffic scenarios.

REFERENCES

Cagienard, R., Grieder, P., Kerrigan, E., and Morari, M. (2007). Move blocking strategies in receding horizon control. *Journal of Process Control*, 17(6), 563–570.

Christidis, P. and Rivas, J. (2012). Measuring road congestion. *Institute for Prospective Technological Studies.*

Hennessy, D. and Wiesenthal, D. (1999). Traffic congestion, driver stress, and driver aggression. *Aggressive*

Table 2. The TTS, mean and max computation time, and the relative change in TTS and mean computation time with respect to the conventional MPC controller for the different demand scenarios and controllers.

Demand scenario	Controller	TTS [veh · h]	TTS^{rel} [%]	Mean computation time [s]	Max computation time [s]	CT^{rel} [%]
High-demand	Fixed times	495.8	106	-	-	-
	MPC	240.5	-	2065	2166	-
	PMPC - relative queue lengths	245.8	2.20	166.6	312	-91.9
	PMPC - Grammatical Evolution	276.6	15.0	168	269	-91.9
Two-peaks	Fixed times	396.2	81.3	-	-	-
	MPC	218.5	-	2041	2167	-
	PMPC - relative queue lengths	220.2	0.77	158	347	-92.2
	PMPC - Grammatical Evolution	244.4	11.9	189	292	-90.7
Alternating-demand	Fixed times	360.2	69.7	-	-	-
	MPC	212.2	-	2118	2166	-
	PMPC - relative queue lengths	216.0	1.78	157	249	-92.6
	PMPC - Grammatical Evolution	228.4	7.63	164	346	-92.2

Table 3. The TTS and relative change in TTS for the different demand scenarios and controllers for which the optimization is terminated after 60 seconds.

Controller	**High-demand**		**Two-peaks**		**Alternating-demand**	
	TTS [veh · h]	TTS^{rel} [%]	TTS [veh · h]	TTS^{rel} [%]	TTS [veh · h]	TTS^{rel} [%]
MPC without time constraint	240.5	-	218.5	-	212.2	-
MPC with time constraint	301.0	25.2	263.9	20.8	276.3	30.2
PMPC - relative queue lengths	247.3	2.83	221.4	1.32	215.7	1.65
PMPC - Grammatical Evolution	272.6	13.3	238.0	8.89	232.7	9.21

Behavior: Official Journal of the International Society for Research on Aggression, 25(6), 409–423.

Hopcroft, J., Motwani, R., and Ullman, J. (2006). *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 3rd Edition.

Jamshidnejad, A., Lin, S., Xi, Y., and De Schutter, B. (2019). Corrections to "Integrated urban traffic control for the reduction of travel delays and emissions". *IEEE Transactions on Intelligent Transportation Systems*, 20(5), 1978–1983. doi:10.1109/TITS.2018.2844465.

Jamshidnejad, A., Papamichail, I., Papageorgiou, M., and De Schutter, B. (2017). Sustainable model-predictive control in urban traffic networks: Efficient solution based on general smoothening methods. *IEEE Transactions on Control Systems Technology*, 26(3), 813–827.

Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2), 87–112.

Krajzewicz, D., Hertkorn, G., Rössel, C., and Wagner, P. (2002). SUMO (Simulation of Urban MObility)-an open-source traffic simulation. In *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, 183–187.

Lin, S., De Schutter, B., Xi, Y., and Hellendoorn, H. (2011). Fast model predictive control for urban road networks via MILP. *IEEE Transactions on Intelligent Transportation Systems*, 12(3), 846–856.

Lin, S., De Schutter, B., Xi, Y., and Hellendoorn, H. (2012). Efficient network-wide model-based predictive control for urban traffic networks. *Transportation Research Part C: Emerging Technologies*, 24, 122–140.

Lin, S. and Xi, Y. (2008). An efficient model for urban traffic network control. *IFAC Proceedings Volumes*, 41(2), 14066–14071.

Nagurney, A. (2000). Congested urban transportation networks and emission paradoxes. *Transportation Research Part D: Transport and Environment*, 5(2), 145–151.

Nicolau, M. and Agapitos, A. (2018). Understanding grammatical evolution: Grammar design. In *Handbook of Grammatical Evolution*, 23–53. Springer.

O'Neill, M. and Ryan, C. (2001). Grammatical evolution. *IEEE Transactions on Evolutionary Computation*, 5(4), 349–358.

Pippia, T., Sijs, J., and De Schutter, B. (2018). A parametrized model predictive control approach for microgrids. In *Proceedings of the 57th IEEE Conference on Decision and Control*, 3171–3176. Miami Beach, Florida.

Poli, R., Langdon, W.B., McPhee, N.F., and Koza, J. (2008). *A Field Guide to Genetic Programming.* Lulu.com.

Rawlings, J. and Mayne, D. (2009). *Model Predictive Control: Theory and Design.* Nob Hill Pub.

Wegener, A., Piórkowski, M., Raya, M., Hellbrück, H., Fischer, S., and Hubaux, J. (2008). TraCI: an interface for coupling road traffic and network simulators. In *Proceedings of the 11th communications and networking simulation symposium*, 155–163.

Ye, B., Wu, W., Li, L., and Mao, W. (2016). A hierarchical model predictive control approach for signal splits optimization in large-scale urban road networks. *IEEE Transactions on Intelligent Transportation Systems*, 17(8), 2182–2192.

Ye, B., Wu, W., and Mao, W. (2015). Distributed model predictive control method for optimal coordination of signal splits in urban traffic networks. *Asian Journal of Control*, 17(3), 775–790.

Ye, B., Wu, W., Ruan, K., Li, L., Chen, T., Gao, H., and Chen, Y. (2019). A survey of model predictive control methods for traffic signal control. *IEEE/CAA Journal of Automatica Sinica*, 6(3), 623–640.

Zegeye, S., De Schutter, B., Hellendoorn, H., Breunesse, E., and Hegyi, A. (2012). A predictive traffic controller for sustainable mobility using parameterized control policies. *IEEE Transactions on Intelligent Transportation Systems*, 13(3), 1420–1429.