Multi-agent Model Predictive Control of Signaling Split in Urban Traffic Networks*

Lucas Barcelos de Oliveira†
Dept. of Automation and Systems Engineering
Federal University of Santa Catarina
Florianópolis, SC 88040-900, Brazil
lucas@das.ufsc.br

Eduardo Camponogara
Dept. of Automation and Systems Engineering
Federal University of Santa Catarina
Florianópolis, SC 88040-900, Brazil
camponog@das.ufsc.br

ABSTRACT

Urban traffic networks are large, dynamic systems which remain a challenge in control engineering despite all of the scientific and technological progress. The sheer size, wide spread of sensors and control devices, and nonlinearities make such systems complex, beyond the scope of existing models, let alone control algorithms. To this end, control engineers have looked for unconventional means for modeling and control, in particular the technology of multi-agent systems whose appeal stems from their composite nature, flexibility, and scalability. This paper contributes to this evolving technology by proposing a framework for multi-agent control of linear, dynamic systems. The framework decomposes a centralized model predictive control problem into a network of coupled, but small sub-problems that are iteratively solved by the distributed agents. Theoretical results ensure convergence of the distributed iterations to a globally optimal solution. The framework is applied to the signaling split control of traffic networks. Experiments conducted with simulation software indicate that the multi-agent framework attains performance comparable to conventional control, such as the traffic-responsive urban control strategy.

Categories and Subject Descriptors

F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence; J.6 [**Computer-aided Engineering**]

Keywords

Urban traffic networks, split control, distributed agents, model predictive control

1. INTRODUCTION

Much of the improvements in urban traffic control can be attributed to past advances in science and technology. The existing technology is changing the way traffic systems are designed and operated. Today, modern operating centers receive traffic-flow data from distributed sensors and implement control policies in response to the prevailing traffic conditions. Yet, there is room for further improvements in urban traffic control to cope with the increasing volume of traffic which incurs pollution, excessive fuel consumption, and prolonged journey times.

Over the last few years, the **T**raffic-responsive **U**rban **C**ontrol (TUC) strategy has drawn attention for its robustness and good performance, specially so under saturated traffic conditions [11]. Such results have been corroborated in field applications in cities as Munich, Glasgow, Southampton, and Chania [3, 12, 16]. The TUC framework models traffic flow using a variation of the store-and-forward model originally proposed in [13], which uses purely continuous state and control variables allowing the computation of control policies with efficient algorithms. In its standard form, TUC calculates the control signals with a two-stage multi-variable regulator [11]: the first stage solves an unconstrained linear-quadratic-regulator (LQR) problem that minimizes a quadratic function on queue lengths and control signals; the second stage recovers feasibility of the control signals produced by LQR, whereby an optimization problem is solved to minimize the distance of the infeasible solution to the feasible space. However, such two-stage procedure does not guarantee optimality [4]. To this end, a model predictive control (MPC) approach was proposed to explictly handle the constraints, this way guaranteeing control feasibility and improving solution quality [9].

Alongside the progress on traffic-flow modeling and control, a great deal of research has advanced the technology of multi-agent systems, notably in the fields of artificial intelligence and software engineering [15, 18]. This evolving technology seeks to assemble agents of limited knowledge and abilities in a multi-agent organization to perform tasks that are beyond the expertise of its individual members. Such agents not only encapsulate information, but they also exhibit semi-autonomous behavior by employing some form of reasoning to cooperate with others for the interest of the whole organization, negotiate to resolve conflicts, and even compete when driven by self-interest. The problem-solving ability of a multi-agent system emerges from the interactions and collective effort of the agents, not only their intelligent behavior.

Multi-agent frameworks were originally restricted to the field of computer science where typical applications are of discrete nature such as puzzle solving, planning, and combinatorial arrangement. More recently, control engineers realized that such frameworks can be extended to operate dynamic systems, specially complex distributed systems such as petrochemical plants and transportation networks [19, 22, 23]. The operation of such complex, spatially-distributed dynamic systems is a formidable challenge to control engineering, to a great extent due to the intrinsic complexity, sheer size, and nonlinearities. Control engineers have turned their atten-

*This research was supported in part by Conselho Nacional de Pesquisa e Desenvolvimento Tecnológico (CNPq) under grants 551050/2005-5 and 473841/2007-0.
†Supported by CNPq Fellowship Program.

tion to multi-agent systems whose appeal stems from their composite nature, flexibility, and scalability [24].

However, multi-agent systems are still a long way from delivering this promise to complex dynamic systems. Much of the literature offers methodologies, general guidelines, or otherwise ad hoc procedures lacking formal methods that ensure convergence and stability. To this end, this paper proposes a framework for controlling linear dynamic systems with a network of distributed control agents. These dynamic systems arise from the interconnection of linear sub-systems with local input constraints. Applications are found in signaling split control in traffic networks and reaction control in petrochemical plants. Given dynamic equations and algebraic constraints, our framework formulates the optimization problem arising from model predictive control and proceeds to decompose the MPC problem into a network of coupled, but small sub-problems to be solved by the agent network. An agent senses only the state variables and sets the values of the control variables of its sub-system, communicating with agents in the vicinity to obtain the values of neighborhood variables and coordinate their actions. With a well-crafted problem decomposition and coordination protocol, the solution iterates produced by the agents can be shown to converge to a globally optimal solution to the MPC problem.

In essence, the decision-making and cooperative control behavior of the agents emerges from the solution of optimization problems. The work reported here builds upon preceding work on distributed control [6, 8] by exploiting the linear dynamic structure to develop simpler models and algorithms.

In a representative traffic network, computational experiments are conducted to assess the performance of the proposed multi-agent MPC framework. The purpose of the experiments is twofold. First, the experiments compare a single, centralized agent with a network of distributed control agents at solving the MPC problem for a number of initial conditions. Second, they compare the multi-agent approach with the TUC strategy using metrics provided by a professional simulation package.

The remaining sections are structured as follows. Section 2 offers some basic concepts about urban traffic networks, along with a description of the store-and-forward model of traffic flow and the LQR strategy used by the TUC approach. Section 3 formulates the MPC problem for split control as a linear dynamic system consisting of a network of dynamically coupled sub-systems, one for each intersection. Last but not least, the section develops a perfect decomposition of the MPC problem into a network of sub-problems and outlines a distributed algorithm for the agent network, which can be shown to converge to an optimal solution. Section 4 reports results from numerical analyses designed to compare the centralized and distributed solution of the MPC problem and from simulated experiments aimed to compare the TUC LQR strategy with the multi-agent MPC approach. Section 4 makes some final remarks and suggests directions for future research.

2. URBAN TRAFFIC CONTROL

An **U**rban **T**raffic **N**etwork (UTN) comprises a set of roads, arterials and streets, known as *links*, interconnected by *junctions* that may be controlled [11]. The traffic inside the network is divided into *streams* of vehicles. Streams grouped in a same link define its *saturation flow*, which is the mean flow crossing the stop line of an approach when the respective stream has the right of way (r.o.w.), a sufficiently large upstream queue, and unobstructed downstream links. The repeated sequence of signal combinations at a junction is named *signal cycle*. Its duration is called *cycle time* or simply *cycle*. A *stage*, or *phase*, is a portion of a signal cycle in which a set of streams has the r.o.w. For safety measures, stages are interposed by constant *lost times* of a few seconds avoiding interference amongst conflicting streams (Fig. 1).

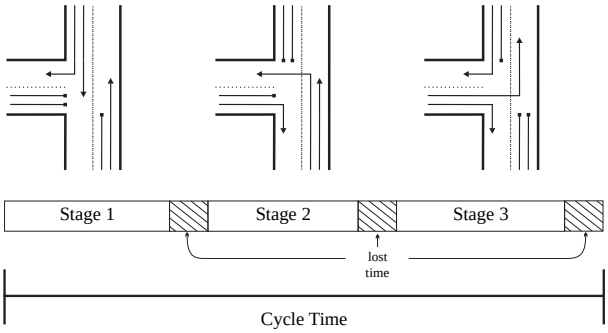

Figure 1: Signal cycle, lost time and cycle.

The influence of traffic lights on traffic depends on four factors [10, 20]: stage specification, cycle duration, offset among junctions, and signaling split. Where split refers to the relative green percentage of the cycle time assigned to each stage.

2.1 UTN Modeling

A UTN modeled in accordance with the TUC strategy [11, 12] is represented as a directed graph with links $z \in Z$ and junctions $j \in J$. Sets I_j and O_j denote, respectively, the incoming and outgoing links of junction j. Cycle times C_j, lost times L_j, turning rates $t_{z,w}$, $z \in I_j$, $w \in O_j$, and saturation flows S_z, $z \in I_j$, are considered constant and known. For the sake of simplicity, $C = C_j$ is assumed for all junctions $j \in J$. Finally, the control signal of junction j has a fixed number of stages belonging to the set F_j, where subset $v_z \subseteq F_j$ represents those where link z has the r.o.w.

Letting $u_{j,i}$ denote the green time of phase i at junction j, the constraint $\sum_{i \in F_j} u_{j,i} + L_j = C$ must be enforced. Additionally, $u_{j,i} \in [u_{j,i}^{min}, u_{j,i}^{max}]$ where $u_{j,i}^{min}$ and $u_{j,i}^{max}$ are the minimum and maximum allowable green times, respectively.

The main differential of this strategy is the use of a variation of the store-and-forward model, where the control cycle is required to be greater than every cycle of the network. Therefore traffic flow is modeled as purely continuous, allowing the use of efficient algorithms on the control signal computation.

The dynamics of network link z is given by equation:

$$\Delta x_z(k+1) = T[q_z(k) + d_z(k) - f_z(k) - s_z(k)], \qquad (1)$$

where: x_z denotes the number of vehicles in link z; q_z and f_z are, respectively, the inflow and outflow of link z during period $[kT, (k+1)T]$, where $k = 1, 2, \ldots$ is a discrete time index and T is the control interval; d_z is the demand, vehicles entering the network not originating from adjacent links; and, finally, s_z is the exit flow at time k.

Since exit rates are known, the exit flow may be replaced for the following equality: $s_z(k) = t_{z,0} q_z(k)$. In addition, one may formulate the inflow of link z as $q_z(k) = \sum_{w \in I_j} t_{w,z} f_w(k)$, where $t_{w,z}$ is the turning rate towards link $z \in O_j$ coming from link $w \in I_j$. Assuming that inflows and outflows of link z with r.o.w. are equal to their saturation flow, S_z, equation (1) is written as:

$$x_z(k+1) = x_z(k) + T\left[d_z(k) - \frac{S_z}{C}\sum_{i \in v_z} u_{j',i}(k) + (1 - t_{z,0}) \sum_{w \in I_j} \frac{t_{w,z} S_w}{C} \sum_{i \in v_w} u_{j,i}(k)\right], \qquad (2)$$

where the control signal $u_{j,i}(k)$ is the green time for vehicles going through junction j during phase i, whereas $\sum_{i \in v_z} u_{j',i}(k)$ is the green time for vehicles leaving link z. Notice that z leaves junction j and enters j'. Generalizing equation (2) for all network links leads to the matrix equation:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k) + T\mathbf{d}(k), \tag{3}$$

where: $\mathbf{x}(k)$ is the state vector; $\mathbf{u}(k)$ is the control vector containing signals $u_{j,i}$, $\forall i \in F_j$, $\forall j \in J$; $\mathbf{d}(k)$ is the vector containing demands d_z, $\forall z \in Z$; and $A = I$, B, and T are the state, input, and disturbance matrices, respectively.

2.2 Split Control

In a traffic-responsive control strategy the signaling split must be optimized according to the demands of involved streams. In standard form, the TUC strategy uses the LQR theory to find an efficient time-invariant gain matrix, which is simpler than optimizing a physical criterion [11] but invariably achieving a sub-optimal control law. By assuming $\mathbf{d}(k) = 0$, the dynamic system (3) becomes:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k), \tag{4}$$

allowing the application of the LQR methodology. The control law thereof does not account for feedforward terms, which is plausible since the main goal is to attain a satisfactory gain matrix rather than an optimized criterion.

Intending to minimize the risk of oversaturation and spillback, minimization of proportional occupancy of links is attempted, i.e. x_z/x_z^{max}, where x_z^{max} is the capacity of link z. A quadratic criterion to this end has the form:

$$\mathcal{J} = \frac{1}{2}\sum_{k=0}^{\infty}(\|\mathbf{x}(k)\|_Q^2 + \|\mathbf{u}(k)\|_R^2), \tag{5}$$

where Q and R are diagonal positive weighting matrices, respectively semi-definite and definite. According to the LQR theory, an infinite time horizon is used in (5) to achieve a time-invariant control law. As matrix Q weighs the states, that is, the number of vehicles in the roads, the goal of minimizing the average occupancy is obtained by making its diagonal elements equal to $1/(x_z^{max})^2$, for the corresponding link $z \in Z$. Matrix R reflects the penalty imposed on control effort, usually defined as $R = rI$, where r is found experimentally.

Minimizing criterion (5) leads to the control law:

$$\mathbf{u}(k) = \mathbf{u}^N - L\mathbf{x}(k), \tag{6}$$

where: $\mathbf{u}(k)$ is the vector with green times $u_{j,i}, \forall j \in J, \forall i \in F_j$; $\mathbf{u}^N$ is the matching vector containing the nominal green times; and L is Ricatti's gain matrix, depending on A, B, Q, and R, though with small susceptibility to their variation [11].

As control constraints are not considered in the aforementioned control law, they are imposed in an ad hoc manner, through the following optimization problem for each junction $j \in J$:

$$\min_{U_{j,i}} \quad \sum_{i \in F_j} (u_{j,i} - U_{j,i})^2 \tag{7a}$$

s. to:

$$\sum_{i \in F_j} U_{j,i} + L_j = C_j \tag{7b}$$

$$U_{j,i} \in [u_{j,i}^{min}, u_{j,i}^{max}],\ \forall i \in F_j, \tag{7c}$$

where $U_{j,i}$ is the closest feasible solution in Euclidean space to $u_{j,i}$.

This problem is solved in real-time for each junction j with an efficient algorithm [10], whose convergence is guaranteed in a number of steps less than or equal to the number of stages $|F_j|$ of the junction. Though this approach gives a feasible solution, it does not satisfy the optimality conditions for system (4). Additionally, because no predictions are made, the multivariable regulator behaves in a purely reactive way to unknown disturbances. On the other hand, the structure of matrix L provides the regulator a gating effect preventing oversaturation in downstream links.

Previously published works [1, 9] report that significant improvement may arise from the replacement of the usual LQR procedure with a solution that accounts for system constraints, such as a model predictive control strategy. Generally speaking, a model predictive control approach is composed by [4, 17]:

- a *prediction model* describing satisfactorily the process dynamics in a finite time horizon;
- a *cost function* which gives the control signal when minimized; and
- a *sliding horizon* of prediction and control, which is translated a step forward at each sample period, requiring the computation of new control actions from which only that of the actual time is implemented.

Following these premises the MPC problem for split control is cast as:

$$P: \min \quad \sum_{k=1}^{T} \frac{1}{2}[\mathbf{x}(k)^T Q\mathbf{x}(k) + \mathbf{u}(k-1)^T R\mathbf{u}(k-1)] \tag{8a}$$

s. to: $\forall k \in \mathcal{T}$:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k) \tag{8b}$$

$$C\mathbf{u}(k) \geq \mathbf{c} \tag{8c}$$

$$D\mathbf{u}(k) = \mathbf{d} \tag{8d}$$

where: $\mathbf{x}(k)$ is the system's state and $\mathbf{u}(k)$ the control input at time k; Q is positive semi-definite and R positive definite weighting matrices; C and $\mathbf{c}$ define the inequality constraints; D and $\mathbf{d}$ define the equalities; and $\mathcal{T} = \{0, \ldots, T-1\}$ is the time horizon.

3. MULTI-AGENT MPC

This section develops an MPC formulation for systems consisting of the interconnection of linear dynamic sub-systems with local constraints, hereafter called linear dynamic networks. The split control problem is cast as an MPC problem over a linear dynamic network, where sub-systems correspond to intersections, state variables correspond to vehicle queues, and control variables represent green times. After the problem formulation, the section presents a decomposition of the MPC problem into a network of coupled, but small sub-problems that are solved iteratively by the agent network. This section reports theoretical properties of the decomposition, relating the MPC problem and sub-problem network, and outlines a distributed protocol to synchronize agent iterations. Conditions are given for the iterations of the agents to arrive at a solution to the centralized MPC problem.

3.1 Modeling and MPC Formulation

The dynamic representation of the traffic-flow derived from the store-and-forward modeling approach is conveniently represented as a system of M interconnected sub-systems, one for each junction. Sub-system m's local state is $\mathbf{x}_m \in \mathbb{R}^{n_m}$ and control signal is $\mathbf{u}_m \in \mathbb{R}^{p_m}$. A directed graph $G = (V, E)$ models the couplings among the sub-systems: an arc $(i, j) \in E$ means that the

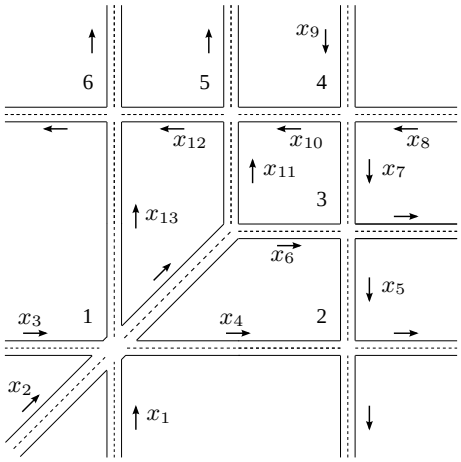

Figure 2: Traffic network.

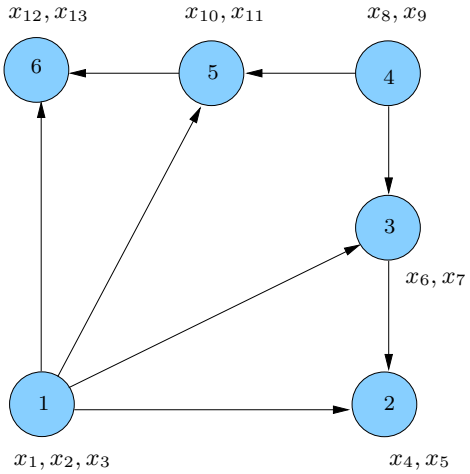

Figure 3: Dynamic coupling graph.

control signals from sub-system i influence the state of sub-system j directly. Assuming discrete-time dynamics, the state equation for sub-system m is:

$$\mathbf{x}_m(k+1) = A_m \mathbf{x}_m(k) + \sum_{i \in I(m)} B_{mi} \mathbf{u}_i(k) \tag{9}$$

where $I(m) = \{m\} \cup \{i : (i,m) \in E\}$ is the set of *input neighbors* of sub-system m including m, that is, the sub-systems affecting the state of m. Given the current state of the network, $\mathbf{x}(0)$, a centralized agent following the MPC strategy would solve the problem below at each sample instant:

$$P: \min \quad \sum_{m=1}^{M} \sum_{k=1}^{T} \frac{1}{2} [\mathbf{x}_m(k)^T Q_m \mathbf{x}_m(k) + \mathbf{u}_m(k-1)^T R_m \mathbf{u}_m(k-1)] \tag{10a}$$

$$\text{s. to: } \forall m \in \mathcal{M},\ k \in \mathcal{T}:$$

$$\mathbf{x}_m(k+1) = A_m \mathbf{x}_m(k) + \sum_{i \in I(m)} B_{mi} \mathbf{u}_i(k) \tag{10b}$$

$$C_m \mathbf{u}_m(k) \geq \mathbf{c}_m \tag{10c}$$

$$D_m \mathbf{u}_m(k) = \mathbf{d}_m \tag{10d}$$

where: $\mathbf{x}_m(k)$ is the state of sub-system m at time k and $\mathbf{u}_m(k)$ is its control input; Q_m is positive semi-definite and R_m is positive definite; C_m and $\mathbf{c}_m$ define the inequality constraints; D_m and $\mathbf{d}_m$ define the equalities; and $\mathcal{M} = \{1, \ldots, M\}$ is the set with the indices of the sub-systems.

The test bed is the traffic network depicted in Fig. 2 with 13 one-way roads and 6 junctions. Sub-system 3 has state $\mathbf{x}_3 = (x_6, x_7)$ with the number of vehicles in roads 6 and 7, while the control vector is $\mathbf{u}_3 = (u_6, u_7)$ with the green time for each road. The coupling graph G appears in Fig. 3. The set of input neighbors to sub-system 3 is $I(3) = \{1, 3, 4\}$. Matrix B_{33} expresses the discharge of queue $\mathbf{x}_3$ as a function of green times $\mathbf{u}_3$, while B_{31} (B_{34}) expresses how queue $\mathbf{x}_3$ builds up as $\mathbf{x}_1$ ($\mathbf{x}_4$) is emptied. The inequality constraints impose minimum and maximum green times on the phases. The equalities state that the total green time plus lost time (yellow time) must add up to cycle time. This is a rough explanation of the store-and-forward model proposed in [11, 21].

Notice that sub-system m's state at time k is a function of initial state and control signals prior to time k:

$$\mathbf{x}_m(k) = A_m^k \mathbf{x}_m(0) + \sum_{l=1}^{k} \sum_{i \in I(m)} A_m^{l-1} B_{mi} \mathbf{u}_i(k-l)$$

By using this relation, collecting the control variables in vector $\bar{\mathbf{u}}_m = (\bar{\mathbf{u}}_m(0), \ldots, \bar{\mathbf{u}}_m(T-1))$, and dropping the constant term from the objective [5], P becomes:

$$P: \ \min \quad f(\bar{\mathbf{u}}) = \frac{1}{2} \sum_{m \in \mathcal{M}} \sum_{i \in I(m)} \sum_{j \in I(m)} \bar{\mathbf{u}}_i^T H_{mij} \bar{\mathbf{u}}_j + \sum_{m \in \mathcal{M}} \sum_{i \in I(m)} \mathbf{g}_{mi}^T \bar{\mathbf{u}}_i \tag{11a}$$

$$\text{s. to: } \bar{C}_m \bar{\mathbf{u}}_m \geq \bar{\mathbf{c}}_m,\ m \in \mathcal{M} \tag{11b}$$

$$\bar{D}_m \bar{\mathbf{u}}_m = \bar{\mathbf{d}}_m,\ m \in \mathcal{M} \tag{11c}$$

where H_{mij}, $\bar{C}_m$, and $\bar{D}_m$ are suitable matrices and $\mathbf{g}_{mi}$, $\bar{\mathbf{c}}_m$, and $\bar{\mathbf{d}}_m$ are suitable vectors. Here, the issue is how a network of distributed agents solves P instead of a centralized agent. In what follows, we develop a decomposition of P into a set of coupled sub-problems $\{P_m\}$ and outline a distributed solution protocol.

3.2 Multi-agent Distributed Control

In our framework for multi-agent control, an agent m decides upon the values of $\bar{\mathbf{u}}_m$ to control sub-system m. For the problem decomposition to be perfect, each agent m solves a local optimization problem P_m encompassing all the terms of f and constraints that depend on $\bar{\mathbf{u}}_m$. Let:

- $\bar{I}(m) = \{i : m \in I(i),\ i \neq m\}$ be the set of *output neighbors* of sub-system m;
- $C(m) = \{(i,j) \in I(m) \times I(m) : i = m \text{ or } j = m\}$ be the sub-system pairs of quadratic terms in Φ_m that depend on $\bar{\mathbf{u}}_m$;
- $C(m,k) = \{(i,j) \in I(k) \times I(k) : i = m \text{ or } j = m\}$ be the pairs of quadratic terms in Φ_k, $k \in \bar{I}(m)$, that depend on $\bar{\mathbf{u}}_m$.

In the traffic network, $I(1) = \{1\}$, $\bar{I}(1) = \{2, 3, 5, 6\}$, $C(1) = \{(1,1)\}$, and $C(1,3) = \{(1,3), (1,4), (1,1), (3,1), (4,1)\}$. Notice that $\bar{\mathbf{u}}_m$ appears in sub-systems $i \in I(m) \cup \bar{I}(m)$, but can

be coupled to other sub-systems—sub-system 1 is coupled to sub-system 4 via sub-system 3, but $4 \notin I(1) \cup \bar{I}(1)$. The notion of neighborhood will establish the interdependence among sub-systems. Models and algorithms for imperfect problem decomposition are found in [7]. According to agent m's view of the system, the control variables are divided in three sets:

- *local variables:* the variables in vector $\bar{\mathbf{u}}_m$;
- *neighborhood variables:* all the variables in vector $\bar{\mathbf{y}}_m = (\bar{\mathbf{u}}_i : i \in N(m))$ where $N(m) = I(m) \cup \{i : (i,j) \in C(m,k), k \in \bar{I}(m)\} - \{m\}$ is the neighborhood of agent m. Notice that $\bar{I}(m) \subseteq N(m)$.
- *remote variables:* the other variables which consist of vector $\bar{\mathbf{z}}_m = (\bar{\mathbf{u}}_i : i \notin N(m) \cup \{m\})$.

According to the perfect decomposition, $P_m(\bar{\mathbf{y}}_m)$ is obtained from P by i) discarding from the objective f the terms not involving $\bar{\mathbf{u}}_m$ and ii) dropping the constraints not associated with agent m. More formally, agent m's local problem is:

$$P_m(\bar{\mathbf{y}}_m): \quad \min \quad f_m = \frac{1}{2}\bar{\mathbf{u}}_m^T H_m \bar{\mathbf{u}}_m + \mathbf{g}_m^T \bar{\mathbf{u}}_m \tag{12a}$$

$$\text{s. to:} \quad \bar{C}_m \bar{\mathbf{u}}_m \geq \bar{\mathbf{c}}_m \tag{12b}$$

$$\bar{D}_m \bar{\mathbf{u}}_m = \bar{\mathbf{d}}_m \tag{12c}$$

where H_m is a suitable matrix and $\mathbf{g}_m$ is a vector. For each agent m, the perfect decomposition ensures that:

$$f(\bar{\mathbf{u}}) = f_m(\bar{\mathbf{u}}_m, \bar{\mathbf{y}}_m) + \bar{f}_m(\bar{\mathbf{y}}_m, \bar{\mathbf{z}}_m)$$

for a given function $\bar{f}_m$. Hereafter $\{P_m(\bar{\mathbf{y}}_m)\}$ will denote the set of sub-problems for all $m \in \mathcal{M}$.

3.2.1 Properties

Below, we report some properties relating P and $\{P_m(\bar{\mathbf{y}}_m)\}$ which are useful to design a distributed algorithm for the agent network. Demonstrations and illustrations are found in [5].

PROPOSITION 1. *A solution $\bar{\mathbf{u}}$ satisfies first-order optimality (KKT) conditions for P if, and only if, $(\bar{\mathbf{u}}_m, \bar{\mathbf{y}}_m)$ satisfies KKT conditions for $P_m(\bar{\mathbf{y}}_m)$ for each $m \in \mathcal{M}$.*

DEFINITION 1. (Feasible Spaces) *The feasible spaces are:*

- *$U_m = \{\bar{\mathbf{u}}_m : \bar{C}_m \bar{\mathbf{u}}_m \geq \bar{\mathbf{c}}_m,\ \bar{D}_m \bar{\mathbf{u}}_m = \bar{\mathbf{d}}_m\}$ is the feasible space for $P_m(\bar{\mathbf{y}}_m)$;*
- *$U = U_1 \times \cdots \times U_M$ is the feasible space for P; and*
- *$Y_m = \times_{i \in N(m)} U_i$ is the feasible space for the neighborhood variables of agent m.*

ASSUMPTION 1. (Compactness) *The feasible space, U, is a compact set.*

ASSUMPTION 2. (Strict Feasibility) *There exists $\bar{\mathbf{u}} \in U$ such that $\bar{C}_m \bar{\mathbf{u}}_m > \bar{\mathbf{c}}_m$ and $\bar{D}_m \bar{\mathbf{u}}_m = \bar{\mathbf{d}}_m$ for all $m \in \mathcal{M}$.*

Compactness is a plausible assumption since control signals are invariably bounded. So is the strict feasibility assumption: if the interior of U is empty, then some inequalities are indeed equalities and should be regarded as such.

PROPOSITION 2. *Problem P given by* (11a)–(11c) *is convex.*

COROLLARY 1. *Sub-problem $P_m(\bar{\mathbf{y}}_m)$ is convex.*

PROPOSITION 3. (Optimality Conditions) *[2] Because f is a convex function and U is a convex set, $\bar{\mathbf{u}}^\star$ is a local minimum for f over U if and only if:*

$$\nabla f(\bar{\mathbf{u}}^\star)^T (\bar{\mathbf{u}} - \bar{\mathbf{u}}^\star) \geq 0,\ \forall \bar{\mathbf{u}} \in U \tag{13}$$

A point $\bar{\mathbf{u}}^\star$ satisfying condition (13) is called *stationary point*.

COROLLARY 2. (Local Optimality Conditions) *$\bar{\mathbf{u}}^*$ is a local minimum for P if, and only if, $(\bar{\mathbf{u}}^*_m, \bar{\mathbf{y}}^*_m)$ is a local minimum for $P_m(\bar{\mathbf{y}}^\star_m)$ for all $m \in \mathcal{M}$.*

A control vector that cannot be improved unilaterally by a single agent, a fixed point, is locally optimal for all the sub-problems and therefore optimal for P.

3.2.2 Distributed Agent Solution

In what follows, we outline a distributed algorithm for the agent network to arrive at a stationary solution to $\{P_m\}$. The agents follow an iterative protocol whereby $\bar{\mathbf{u}}^{(k)} = (\bar{\mathbf{u}}_1^{(k)}, \ldots, \bar{\mathbf{u}}_M^{(k)})$ denotes the solution at iteration k. Starting with a feasible control vector $\bar{\mathbf{u}}^{(0)}$, the agents exchange information locally, synchronize their computations to preclude coupled agents from acting simultaneously, and iterate until convergence is attained.

ASSUMPTION 3. (Synchronous Work) *If an agent m revises its decisions at iteration k, then:*

(i) agent m uses $\bar{\mathbf{y}}_m^{(k)} = (\bar{\mathbf{u}}_i^{(k)} : i \in N(m))$ to obtain an approximate solution to $P_m(\bar{\mathbf{y}}_m^{(k)})$ which becomes $\bar{\mathbf{u}}_m^{(k+1)}$;

(ii) all the agents in the neighborhood of agent m keep their decisions at iteration k, that is, $\bar{\mathbf{u}}_i^{(k+1)} = \bar{\mathbf{u}}_i^{(k)}$ for all $i \in N(m)$.

ASSUMPTION 4. (Continuous Work) *If $\bar{\mathbf{u}}^{(k)}$ is not a stationary point for all problems in $\{P_m\}$, then at least one agent m changes its decisions from $\bar{\mathbf{u}}_m^{(k)}$ to $\bar{\mathbf{u}}_m^{(k+1)}$ by approximately solving $P_m(\bar{\mathbf{y}}_m^{(k)})$ such that $\bar{\mathbf{u}}_m^{(k)}$ is not a stationary point to P_m.*

Condition (ii) of Assumption 3 and Assumption 4 are ensured if the agents iterate in a sequence $\langle S_1, \ldots, S_r \rangle$ where $S_i \subseteq \mathcal{M}$, $\cup_{i=1}^r S_i = \mathcal{M}$, and all distinct pairs $m, n \in S_i$ are non-neighbors for all i. $\langle S_1, S_2, S_3 \rangle$ is such a sequence for the illustrative scenario with $S_1 = \{2, 4, 6\}$, $S_2 = \{3, 5\}$, and $S_3 = \{1\}$. Time-varying sequences and synchronization protocols are alternatives.

Another key issue is how an agent m solves P_m approximately, as stated in condition (i) of Assumption 3, so that $\bar{\mathbf{u}}^{(k)}$ converges to a stationary point for $\{P_m\}$. To this end, we developed an algorithm based on the feasible direction method, which is fully developed in [5] and outlined below. Related frameworks and algorithms for other settings appeared in [6, 8].

At the current iterate $\bar{\mathbf{u}}^{(k)}$, agent m computes a *locally descent direction* $\bar{\mathbf{d}}_m^{(k)} = \hat{\mathbf{u}}_m^{(k)} - \bar{\mathbf{u}}_m^{(k)}$ by solving a linear programming (LP) problem that minimizes $\nabla f_m(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)})^T (\hat{\mathbf{u}}_m^{(k)} - \bar{\mathbf{u}}_m^{(k)})$ subject to the original constraints imposed on the decisions of agent m. It then produces the next iterate $\bar{\mathbf{u}}_m^{(k+1)} = \bar{\mathbf{u}}_m^{(k)} + \alpha_m^{(k)} \bar{\mathbf{d}}_m^{(k)}$ by finding a step $\alpha_m^{(k)}$ that satisfies the *Armijo rule*. Given $(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)}) \in U_m \times Y_m$, $\bar{\mathbf{d}}_m^{(k)} \neq 0$ is a *locally feasible direction* at $(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)})$ if $\bar{\mathbf{u}}_m^{(k)} + \alpha_m \bar{\mathbf{d}}_m^{(k)} \in U_m$ for all $\alpha_m > 0$ that are sufficiently small. A locally feasible direction $\bar{\mathbf{d}}_m^{(k)}$ at a nonstationary point $(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)})$ is a *locally descent direction* if $\nabla f_m(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)})^T \bar{\mathbf{d}}_m^{(k)} < 0$.

Assumption 3, Assumption 4, and agent iterations as delineated above—which use a locally descent direction obtained by solving

an LP problem and satisfy the Armijo rule—ensure that $\bar{\mathbf{u}}^{(k)}$ arrives at a stationary point of $\{P_m\}$ and, thereby, a solution to P. Effectively, the agent network implements a distributed feasible direction method for quadratic programming.

The constraint structure in split control admits simplifications in the iterative processes of the agents. For each junction j and phase i, suppose the maximum green time $u_{j,i}^{max}$ is $C_j - L_j - \sum_{i \in F_j} u_{j,i}^{min}$. Then the MPC problem P, given by (11a) through (11c), can be recast using control variables $\Delta\bar{\mathbf{u}}_m(k)$ with green times in excess to the minimum, namely $\Delta u_{j,i} = u_{j,i} - u_{j,i}^{min}$. This variable change simplifies the inequality constraints (11b) which become simple bounds of the form $\Delta\bar{\mathbf{u}}_m(k) \geq 0$. As a result, the linear program for computing a locally descent direction is solved analytically: the LP constraint structure consists of block-diagonal equalities (one for each time period) and simple variable bounds; the solution is obtained by examining the coefficients of the gradient $\nabla f_m(\bar{\mathbf{u}}_m^{(k)}, \bar{\mathbf{y}}_m^{(k)})$.

The agents are not limited to using the feasible direction method sketched above. They can apply any quadratic-programming algorithm that meets the Armijo rule or otherwise solves the subproblem up to optimality. The active set and gradient projection methods [2] are candidates to replace the feasible direction algorithm.

4. EXPERIMENTAL STUDIES

This section presents results from the application of the TUC LQR strategy and the multi-agent MPC framework for the signaling split control of the UTN depicted in Fig. 2. While TUC uses equation (4) to model the system and objective (5) to compute a feedback gain matrix, it is more appropriate for the MPC framework to express the control problem in terms of objective (10a) subject to constraints (10b) through (10d), as control signals must lie within bounds and hold constant cycle periods.

4.1 Network Set-up

Additional parameters must be specified to fully model the UTN depicted in Fig. 2, such as saturation flows, turning rates, traffic demands, and exit rates. The simulation environment was further simplified: the exit rates of the network are null; lost times in between phases are four seconds; all offsets are zero; and all network links have equal length so that their occupancy contributes with the same weight in the objective function.

Up to this day, fixed-time signaling is still the most usual type of split control worldwide. Since fixed-time control is not adaptive, drivers *learn* and *predict* network dynamics which induce a matching behavior between drivers and the traffic system. Therefore, from a practical perspective, traffic engineers favor a control policy that penalizes deviation from the nominal fixed-time split, rather than a control strategy formulated in terms of absolute control values.

Table 1 presents the nominal splits of the sample UTN and the other aforementioned parameters—nominal splits were obtained with Webster's procedure. Some turn rate parameters are irrelevant and not presented, namely the ones that do not take part in the inflow of another controlled link—e.g., in link 11, **12**:0.5 means that 50% of that link exit flow enters link 12, while the remaining vehicles take a route outside the scope of the controlled network and are not accounted for. In the simulated analysis, all junctions have a constant cycle of 120 s and have two phases, with the exception of junction 1 which has three phases. The mean inflows in the input links are: $q_1 = 800$ veh/h; $q_2 = 1300$ veh/h; $q_3 = 900$ veh/h; $q_8 = 900$ veh/h; and $q_9 = 700$ veh/h.

Table 1: Network specification.

Link (z)	Sat. Flow (S_z) (veh/h)	Nom. Split (u_z^N) (s)	Turn Rate ($t_{z,w}$) (to link: f%)
1	3600	29	**4**:0.2; **6**:0.05; **11**:0.05; **13**:0.7
2	3600	49	**4**:0.25; **6**:0.3; **11**:0.3; **13**:0.15
3	3600	32	**4**:0.65; **6**:0.05; **11**:0.05; **13**:0.15
4	3600	72	—
5	3600	40	—
6	1800	57	**5**:0.5
7	3600	55	**5**:0.8
8	3600	63	**7**:0.4; **10**:0.6
9	3600	49	**7**:0.6; **10**:0.4
10	3600	60	**12**:0.8
11	1800	52	**12**:0.5
12	3600	55	—
13	3600	57	—

4.2 Numerical Analysis

To validate the proposed multi-agent control framework, a set of MPC problems were solved covering a range of initial conditions. A centralized agent solved the global MPC problem P, while a multi-agent system solved the corresponding sub-problem network $\{P_m\}$ for each of the initial conditions, allowing their performance to be compared. Both approaches used a standard quadratic-programming (QP) algorithm [14, active set method] and the feasible direction method outlined above. For the distributed feasible direction method, experiments showed that the acceptance degree $\sigma = 0.3$ and the step-contraction parameter $\beta = 0.3$ for the Armijo rule induce the best convergence rate in the given scenarios. Notice that the distributed QP approach implicitly satisfies the Armijo rule. A set of ten randomly obtained queues defined the initial conditions in the experiments, whose results appear in Table 2.

As the tolerance of the optimization package used for the centralized QP computation could not be modified, results illustrate the computational effort to reach the actual optimal cost. In other instances we assume that the solution has converged once it is within a 0.1% error margin from the optimal objective, previously computed by the centralized QP algorithm.

The experimental results show a trade-off between the complexity of the algorithm and the number of iterations required for convergence. On the other hand, the distinction between the distributed and centralized approach is small, specially so with respect to the feasible direction method. Most importantly, the numerical results confirm the multi-agent control theory outlined above, satisfying optimality conditions for diverse initial conditions.

4.3 Simulated Analysis

The simulated results are from AIMSUN© replications of the sample UTN. AIMSUN© has a powerful micro-simulator for traffic applications which provides accurate modeling of complex networks. Furthermore, it offers a useful API module with the ability to interface, through Python and C++ routines, with almost any external module that needs access to internal data during simulation run time.

The solution of the optimization problems required by the LQR strategy and the multi-agent MPC used several tools: the PSFL licensed Python 2.5 programming language; the OSI-Approved Open Source numerical package for Python, NumPy; the GNU licensed

Table 2: Computational results for a set of ten initial conditions with 0.1% error tolerance.

	Quadratic Programming				Feasible Direction			
	CPU Time (ms)		Iterations		CPU Time (s)		Iterations	
	Mean	Max	Mean	Max	Mean	Max	Mean	Max
Agent 1	21.9	46.9	3.3	7	2.16	8.12	76	290
Agent 2	15.6	46.9	2.1	4	1.98	8.20	72	290
Agent 3	10.9	46.9	2.8	8	1.85	7.59	69	280
Agent 4	14.1	31.2	3.0	6	1.85	7.50	76	294
Agent 5	1.6	15.6	2.4	5	1.83	7.70	69	280
Agent 6	6.3	46.9	1.2	2	1.81	7.67	67	280
Multi-agent	75.0	156.2	14.8	25	11.58	47.16	425	1710
Centralized	26.6	46.9	3.9	7	13.59	48.09	370.9	1299

Table 3: Average results for ten AIMSUN© replications.

	Travel Time (s/km)		Density (veh/km)	
	Avg.	Std. Dev.	Avg.	Std. Dev.
LQR	182.759	3.928	17.853	0.355
MA-MPC	180.288	3.619	17.564	0.314

CVXOPT optimization package; and the solver MOSEK©.

The simulated scenario had a duration of one hour with the inflow patterns given above. Although inflows have constant mean, vehicles do not necessarily enter the network at a constant rate because the simulator uses an exponential feed algorithm. All frameworks share the same weighting matrices, with state matrix $Q = I$ and control matrix $R = rI$, $r = 0.003$, as is usual in the TUC policy. As mentioned earlier, the store-and-forward model requires the control interval to be greater than any cycle in the network. Following this premise, a control cycle of 200 s was defined for the sample UTN. Furthermore, the multi-agent MPC achieved best results when both the prediction and control horizon were set to a single control step. Table 3 reports the average results covering a set of 10 random initial conditions.

4.4 Discussion

The experimental results show that the proposed multi-agent MPC framework can perform slightly better than the TUC strategy in signaling split of urban traffic networks. Nevertheless, some aspects need further investigation.

The first aspect is the length of the prediction horizon. The fact that wider prediction degrades system performance indicates that the TUC store-and-forward model may not be adequate for prediction, suggesting that a more precise model could improve overall system performance.

Another issue is the effect produced by the weighting matrices, particularly the effect on control signals. The weight chosen in the simulation penalizes control deviation from nominal signals very lightly, allowing drastic changes without incurring substantial cost increase in the objective function. This increases the responsiveness of the control system but, on the other hand, affects the synchronization among consecutive junctions.

Although offset and stage specification were not object of study in this paper, they influence the performance metrics and should be accounted for in simulated analyses. This justifies the design of the sample UTN with only one-way links, which ease the specification of stages. Although such measure increases the reliability of the experimental model, circumventing the distortion caused by the lack of synchronization control and offset dimensioning is another issue to be investigated.

Some considerations regarding the practical implementation of the proposed strategy are pertinent. First, the method requires full knowledge of system state, either through the installation of inductive loop detectors or other means such as image detection devices. Although the exchange of messages is necessary at every control cycle, it does not constrain the application of multi-agent MPC due to the large control interval. The same communication infrastructure used in the centralized control scheme can be used for distributed control with a centralized message relay. Finally, several practical issues should be analyzed in field applications, such as the influence of noise, delays, and poor synchronization.

The slightly better multi-agent MPC performance is further endorsed by other advantages of this approach. First, the multi-agent MPC circumvents the lack of reconfigurability of the TUC strategy [11], as the addition of nodes to the network affects only the sub-systems in the vicinity. Another advantage is the use of more precise traffic-flow models, such as the non-linear representation proposed in [1].

5. SUMMARY AND FUTURE WORK

This work has contributed to the state-of-the-art by proposing a framework for multi-agent control of dynamic systems. The class of systems comprises linear dynamic networks that are assembled by interconnecting dynamically-coupled sub-systems. This representative class encompasses dynamic networks that use the store-and-forward model to represent traffic flow dynamics, conveniently capturing the local couplings between neighboring junctions.

The signaling split control for the store-and-forward model entails solving a constrained, infinite time, linear quadratic regulator problem. The TUC approach obtains a feedback control law with the unconstrained LQR technique by disregarding the constraints on control signals, whose feasibility is recovered by solving a quadratic program. Model predictive control handles constraints in a systematic way by using a finite time, rolling horizon and solving optimization problems on-line. This paper proposed a decomposition of the MPC problem in a set of locally coupled sub-problems that are iteratively solved by a network of distributed agents. Under certain mild conditions and synchronous work, the iterations of the multi-agent control system can be shown to be drawn towards a fixed point that induces a globally optimal solution to the MPC problem. Numerical experiments illustrate the convergent behavior of the multi-agent system and compare its speed with that of an ideal, centralized agent that solves the problem single-handed. Simulated studies corroborate the hypothesis that a control algorithm that handles constraints explicitly can outperform strategies that treat constraints in an ad hoc manner.

The work reported heretofore is in its nascent, opening up a number of opportunities for research and studies with multi-disciplinary contributions across the fields of multi-agent technology, control engineering, and transportation systems. Some directions for research are:

- numerical and simulated studies with very large networks aimed to confirm the potential of the multi-agent MPC framework;
- the formulation and application of new traffic models, representing more accurately the flow of vehicles;
- studies demonstrating the flexibility and scalability of multi-agent systems, such as the reconfiguration of a traffic junction which would demand only local adjustments, involving the junction and its immediate neighboring intersections; and
- the formal extension of the framework to handle constraints on state variables, such as limits on queue lengths.

6. REFERENCES

[1] K. Aboudolas, M. Papageorgiou, and E. Kosmatopoulos. Control and optimization methods for traffic signal control in large-scale congested urban road networks. In *Proceedings of the American Control Conference*, pages 3132–3138, New York, USA, July 2007.

[2] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

[3] C. Bielefeldt, C. Diakaki, and M. Papageorgiou. TUC and the SMART NETS project. *IEEE Transactions on Intelligent Transportation Systems*, pages 55–60, 2001.

[4] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer-Verlag, 2004.

[5] E. Camponogara and L. B. de Oliveira. Distributed optimization for model predictive control of linear dynamic networks. Technical report, UFSC, 2007. http://www.das.ufsc.br/∼camponog/papers/tech-dmpc-tuc-07.pdf.

[6] E. Camponogara, D. Jia, B. H. Krogh, and S. N. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, February 2002.

[7] E. Camponogara and S. Talukdar. Designing communication networks to decompose network control problems. *INFORMS Journal on Computing*, 17(2):207–223, 2005.

[8] E. Camponogara and S. N. Talukdar. Distributed model predictive control: synchronous and asynchronous computation. *IEEE Transactions on Systems, Man, and Cybernetics – Part A*, 37(5):732–745, September 2007.

[9] L. B. de Oliveira and E. Camponogara. Predictive control for urban traffic networks: initial evaluation. In *Proceedings of the 3rd IFAC Symposium on System, Structure and Control*, Iguassu, Brazil, October 2007.

[10] C. Diakaki. *Integrated Control of Traffic Flow in Corridor Networks*. PhD thesis, Department of Production Engineering and Management, Technical University of Crete, Greece, 1999.

[11] C. Diakaki, M. Papageorgiou, and K. Aboudolas. A multivariable regulator approach to traffic-responsive network-wide signal control. *Control Engineering Practice*, 10(2):183–195, 2002.

[12] C. Diakaki, M. Papageorgiou, and partners of the project TABASCO. Urban integrated traffic control implementation strategies. Technical Report Project TABASCO (TR1054), Transport Telematics Office, Brussels, Belgium, September 1997.

[13] D. C. Gazis and R. B. Potts. The oversaturated intersection. In *Proceedings of the Second International Symposium on Traffic Theory*, pages 221–237, 1963.

[14] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, UK, 1981.

[15] N. Jennings. On agent-based software engineering. *Artificial Intelligence*, 117:277–296, 2000.

[16] E. Kosmatopoulos, M. Papageorgiou, C. Bielefeldt, V. Dinopoulou, R. Morris, J. Mueck, A. Richards, and F. Weichenmeier. International comparative field evaluation of a traffic-responsive signal control strategy in three cities. *Transportation Research Part A: Policy and Practice*, 40(5):399–413, 2006.

[17] F. Kühne. Controle preditivo de robôs móveis não holonômicos. Master's thesis, Graduate Program in Electrical Engineering, Federal University of Rio Grande do Sul, Brazil, 2005.

[18] F. P. Maturana, R. J. Staron, and K. H. Hall. Methodologies and tools for intelligent agents in distributed control. *IEEE Intelligent Systems*, 20(1):42–49, 2005.

[19] R. R. Negenborn, B. D. Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: serial versus parallel schemes. To appear in *Engineering Applications of Artificial Intelligence*, 2007.

[20] M. Papageorgiou. Overview of road traffic control strategies. In *Information and Communication Technologies: From Theory to Applications*, pages LIX–LLX, 2004.

[21] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang. Review of road traffic control strategies. *IEEE Proceedings*, 91(12):2043–2067, 2003.

[22] D. Srinivasan and M. C. Choy. Cooperative multi-agent system for coordinated traffic signal control. *IEE Proceedings. Intelligent Transport Systems*, 153(1):41–49, 2006.

[23] E. Tatara, I. Birol, F. Teymour, and A. Çinar. Agent-based control of autocatalytic replicators in networks of reactors. *Computers & Chemical Engineering*, 29:807–815, 2005.

[24] E. Tatara, A. Çinar, and F. Teymour. Control of complex distributed systems with distributed intelligent agents. *Journal of Process Control*, 17:415–427, 2007.