*ECAI 2014*
*T. Schaub et al. (Eds.)*


*doi:10.3233/978-1-61499-419-0-1191*

# Stop-Free Strategies for Traffic Networks: Decentralized On-line Optimization

**Mohamed Tlig**[1] and **Olivier Buffet**[1] and **Olivier Simonin**[2]

**Abstract.**

Traffic management in large networks remains an important challenge in transportation systems. The best approach would be to use existing infrastructure and find a solution to manage the increasing flows of vehicles. Multi-agent systems and autonomous vehicles are today considered as a promising approach to deal with traffic control. In this paper, we propose a two-level decentralized multi-agent system which allows autonomous vehicles crossing the network intersections without stopping. At the first level, we use a control agent at each intersection which (1) lets the vehicles from each road pass alternately, and (2) allows them to optimally regulate their speed in its vicinity. At the second level, each agent coordinates with its neighboring agents in order to optimize the flows inside the network. We evaluate this approach empirically, with a comparison with a more opportunistic First-Come First-Served strategy. Experimental results (in simulation) are presented (measuring energy consumption), showing the advantages and disadvantages of each approach.

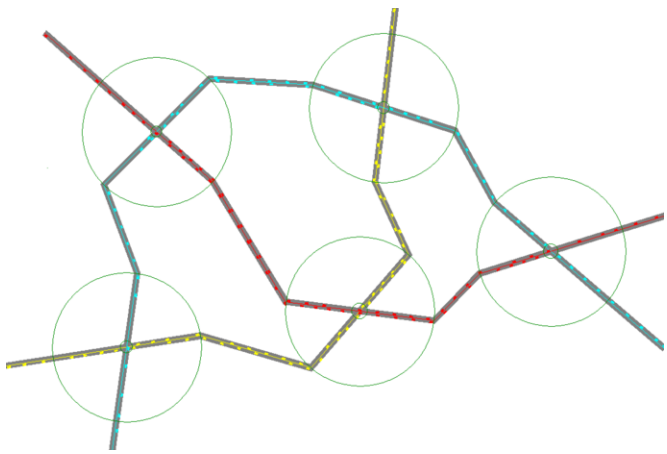

**Figure 1.** Dense traffic conditions in a 3-roads (2-lanes per road) traffic network with 5 intersections

## 1 Introduction

In many real transport systems, congestions are generated at the intersections between the roads [3], i.e., parts of the space which must be shared by the vehicles. There are several methods to manage intersections. The simplest ones generally favor one flow against the other, as traffic lights and "STOP" signals do. Such events generate delays for the vehicles because they require stopping multiple vehicles for some time [5]. If the flow of vehicles is important, these local delays can lead to the emergence of congestions.

There are various ways to handle traffic congestions [1]. A very common approach is to use synchronized traffic lights, i.e., traffic lights which all share the same cycle length (time required for one cycle of traffic light phases), and whose offsets (phases of these periodic signals) are chosen so as to favor "green waves" allowing most vehicles to avoid stopping.

This work has been conducted in the context of the InTraDE European project, in which autonomous vehicles transport containers across a seaport[3]. Yet, we consider generic road networks with multiple intersections, as illustrated on Figure 1. Each autonomous vehicle follows a pre-determined path along one lane. The objective is to reduce delays and energy consumption, and more generally avoid blockings.

Our approach consists in dealing with the traffic at two level: at the local level, i.e., at each intersection, and at the global level, i.e., between intersections. At the first level, the flows are synchronized so that the vehicles can alternately cross the intersection without stopping. This requires (1) adapting the vehicles' speeds so that they arrive at the right time to cross the next intersection without collision, and (2) introducing a control agent at each intersection to handle incoming vehicles. Tlig et al. [15] show how to derive the algorithm in each such control agent and the speed profile for each vehicle as a function of the parameters of the problem (lane and vehicle dimensions, default speed, angle between roads...).

The second level focuses on a key problem: how to optimize the flow throughout the whole network. This requires optimizing the phases (offsets) of the different intersections depending on the chosen criterion, the traffic network, and the traffic conditions (the traffic flow on each lane). This is a potentially high-dimensional optimization problem with as many control variables as there are nodes/intersections in the network. Yet, under mild separability assumptions for the criterion, one can exploit the network structure of the problem to decentralize the optimization process. This decentralization allows to (1) speed up computations and (2) simply perform the optimization on-line while traffic conditions evolve.

We empirically evaluate this approach, taking into account various parameters which come into play, such as the throughput of vehicles or the communication range of the control agent. These experiments are also a means to show how it compares to other approaches, and in particular to another —more opportunistic— stop-free strategy. To that end, we consider two metrics: the total delay accumulated by the vehicles across the network, and the energy consumption due to the speed variations.

[1] INRIA / Université de Lorraine, Nancy, France, email: firstname.lastname@loria.fr
[2] CITI-INRIA, INSA de Lyon, France, email: firstname.lastname@insa-lyon.fr
[3] `http://www.intrade-nwe.eu`

After giving some background in Section 2 on the fields of traffic management and intersections control in general, we describe in Section 3 the two types of stop-free controllers we will consider in the intersections. In Section 4, we show how to optimize the phases for one of these controllers, and how this can be done on-line, in a decentralized manner. Then Section 5 presents an empirical study of this approach, both in off-line and on-line settings. Finally, we discuss the perspectives of this work and conclude.

## 2 Background

In this study, we address the general problem of managing crossing flows of vehicles in road networks. This problem has been traditionally studied in operations research and queueing theory. It typically concerns vehicles driven by humans but, with the arrival of new technologies, many works integrating on-line decisions consider an automatic and real-time control. Several approaches based on communications and GPS (Global Positioning System) propose to improve existing solutions such as traffic lights. In [1], Bazzan presents a summary of the methods and approaches in the fields of traffic engineering to overcome congestion problems at intersections. The author classified these solutions according to the technology used, the reactivity (adaptivity to the traffic-flow variations), and if they are decentralized or not.

**Coordinated systems** are the most deployed systems because they are the oldest one, but also because they are simple and do not require much technology comparing to others. This class contains, e.g., TRANSYT[11] (centralized, not responsive), SCOOT[7] (centralized, responsive), SCATS[8] (decentralized, responsive), PRODYN[6] (decentralized, responsive). They calculate the optimal configuration of the intersections via hill-climbing optimization in order to create "green waves". The major problems we are interested in concerning these kind of approaches are (1) for all these systems the implementation of the traffic lights configuration cannot be done in real time, and (2) when creating "green waves" using traffic lights, we force promoting one flow over others.

**Actuated approaches** are strategies based on physics, optimization, artificial intelligence (Multi-Agent Systems) and learning. For example, in [2] the authors propose an approach based on swarm intelligence and multi-agent systems. Each intersection behaves like a social ant to improve delays and influence its neighbors. The problem with this kind of algorithms is that they take too much time to converge, in addition to the unpredictable behavior of the traffic. In the same context, Wiering in [17] proposes a multi-agent reinforcement learning approach for traffic light control and for vehicles. In this study, each intersection learns a value function in order to estimate expected waiting times given different configurations of traffic lights. Plus, each vehicle uses a value function to compute the optimal trajectory to its destination. This solution is very interesting. However, the complexity of the traffic light problem in conjunction with the complexity of the solution make its application difficult and unrealistic.

**New technologies** Other solutions are interested in fully autonomous vehicle control. They can be classified into two categories.

*Reservation approaches*, introduced by Dresner and Stone in [3, 4], are based on agents managing one intersection each. Each vehicle wanting to cross must book a passage time interval and a route within its target intersection. By doing so, more vehicles can be inside an intersection at the same time —as long as their space-time trajectories do not intersect—, which increases the throughput.

The *decentralized approach* introduced by Rashe and Naumann [9, 10] is based mainly on communication and negotiation between the vehicles to determine the sequence of passage and exit from the intersection. This approach is known for its limits, which depend on the number of vehicles trying to negotiate their passage through the intersection.

## 3 Two Approaches for Stop-Free Intersections

Dresner and Stone's reservation approach [3] allows many vehicles to avoid stopping at intersections by telling them in advance when to arrive. Yet, although this has not been studied up to now, this approach seems quite unlikely to generate green wave phenomena across a network. Indeed, vehicles leave an intersection at a time independent of when they will arrive at the next intersection.

Regular green waves require periodic traffic patterns, what is typically enforced by periodic traffic signals. We thus use the distributed traffic control mechanism proposed in [15], which uses such periodic traffic controllers (analogous in a sense to traffic lights), but also prevents vehicles from stopping.

### 3.1 Alternating

As in [15], we model our traffic network as follows:

- We consider traffic networks made of roads —each with two opposite lanes— and their intersections. The intersections (illustrated by Fig. 2) allow crossing a road but not turning. Two roads intersect at an intersection $i$ and under some angle $\theta_i$.
- Each lane $l$ has its own flow $W_l$.
- The network can thus be seen as a graph $(\mathcal{V}, \mathcal{E})$ where the vertices in $\mathcal{V}$ are intersections and the edges in $\mathcal{E}$ are lane segments $i \to j$, where $i$ and $j$ are neighboring intersections.

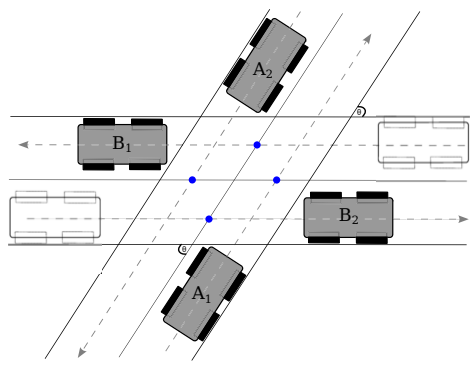


**Figure 2.** An example intersection with two 2-lane roads

Tlig et al. [15] employ a control agent at each intersection $i$ so as to ensure that vehicles from both roads alternately cross the intersection (at default velocity $V$). To that end, an optimal crossing period $T_{min}(i)$ has to be computed as a function of $V$ and of the geometric parameters such as the intersection's angle $\theta_i$. Two vehicles, one from each lane, shall pass at time steps $t = kT_{min}(i)$ for one road ($k \in \mathbb{N}$), and at $t = (k + \frac{1}{2})T_{min}(i)$ for the other. To that end, the control agent of the intersection tells incoming vehicles —as soon as they enter a control zone of radius $R$— when they should enter the crossing area of radius $r_0$ (at speed $V$), which requires them to compute an appropriate speed profile between $R$ and $r_0$ (during which

they temporarily decelerate). Fig. 3 illustrates this alternating principle of half-period $T_c = \frac{T_{min}(i)}{2}$ (the minimum time needed to pass a single vehicle through the intersection).

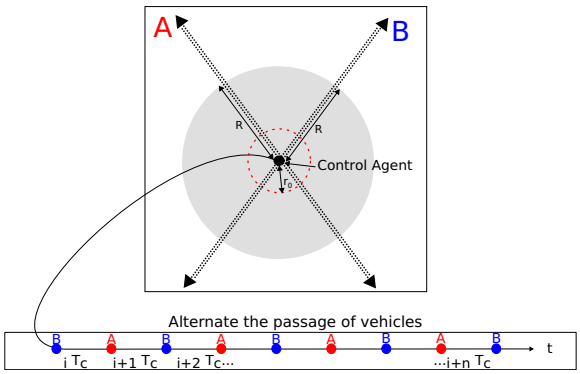


**Figure 3.** Alternating principle of an intersection for two roads $A$ and $B$

## 3.2 First-Come First-Served

The above Alternating (Alt) strategy may waste time at some intersection $i$ when a group of vehicles would like to pass on one lane while no vehicles are going through orthogonal lanes. We thus propose to also consider another strategy which does not constrain to alternate vehicles from one road and the other but —still using the same periodic signal— lets the vehicles book the time when they traverse. This *First-Come First-Served* (FCFS) strategy somehow relates to Dresner and Stone's reservation strategies.

Let us for example consider an intersection $i$ where roads $A$ and $B$ cross each other. Let us call $A_1$, $A_2$, $B_1$ and $B_2$ the lanes corresponding respectively to the roads $A$ and $B$. Fig. 4 (where $T = T_{min}(i)$) illustrates the FCFS approach. Here, the order of arrival of vehicles on the various lanes leads to (1) allocating the first half-period ($\frac{T}{2}$) to a single vehicle from lane $A_1$, which is followed (2) by two vehicles from road $A$ (one on each lane) at the second half-period ($T$), then (3) a vehicle from lane $B_2$ can pass during the third half-period ($\frac{3T}{2}$), and so on.

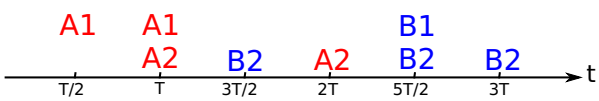


**Figure 4.** Illustration of the First-Come First-Served approach at the intersection of roads $A$ and $B$ showing that only vehicles from the same road —but from opposite lanes (either $A_1$ and $A_2$, or $B_1$ and $B_2$)— can go through the intersection at the same half-period

# 4 How to Create Green Waves

As can be noted, the two control strategies presented in the previous section let each intersection have its own period $T_{min}(i)$. Yet, green waves require that all intersections on the same lane share the same period. This typically implies that all intersections in the network should share the worst period: $T_{max} = \max_{i \in \mathcal{V}} T_{min}(i)$. Note that this does not significantly alter the maximum throughput in the intersections since $T_{min}(i)$ does not vary much when the value of $\theta_i$ remains in the usual range $[\frac{\pi}{3}, \frac{2\pi}{3}]$.

Given the above Alternating approach, the only control variable left is the phase (offset) $\phi_i \in [0, 2\pi)$ of each intersection's periodic signal (which was omitted in the previous section). As shown in Fig. 5, an intersection $i$ directly interacts with its neighbors (here $i$'s neighborhood is $N(i) = \{j, k, o, p\}$), their phases constraining the duration of the traversal of each segment. For example, one can obtain a desired average speed $V_{i,j}$ on segment $i \to j$ by tuning the phases $\phi_i$ and $\phi_j$ appropriately. Yet, each of the 8 segments represents a different constraint to satisfy (/criterion to optimize) while only 5 parameters can be set, meaning that, even on this small example, compromises will be required.

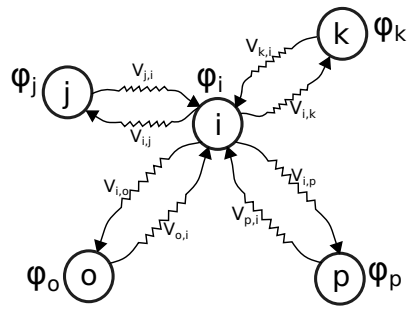


**Figure 5.** The optimal phase $\phi_i$ for intersection $i$ depends on the phases of its neighbors

We now discuss which criteria can be optimized by controlling phases before describing appropriate optimization algorithms.

## 4.1 Separable Optimization Criteria

Let us note $\vec{\phi}$ the vector of all the network intersection phases. We focus here on two objectives, which will be considered independently:

1. minimizing the travel times, i.e., the sum of the travel times $t_{i,j}$ on each lane segment $i \to j$, weighted by the flow $W_{i,j} = W_l$ (where $l$ is the lane corresponding to the segment $i \to j$):

$$t_{global}(\vec{\phi}) = \sum_{(i \to j) \in \mathcal{E}} W_{i,j} t_{i,j}(\vec{\phi}), \text{ and}$$

2. minimizing the total energy consumption, i.e., the sum of the kinetic energy $E_{i,j}$ on each lane segment $i \to j$, weighted by the flow $W_{i,j}$:

$$E_{global}(\vec{\phi}) = \sum_{(i \to j) \in \mathcal{E}} W_{i,j} E_{i,j}(\vec{\phi}).$$

Note that there is no risk that such a minimization leads to blocking certain vehicles since we assume that predefined periodic traffic signals are used that allow all vehicles to pass. This would be different for example if optimizing the duration of red and green phases of traffic lights.

As can be observed, under mild assumptions, both criteria described above are additively separable in that they can be written using as follows:

$$f_{global}(\vec{\phi}) = \sum_{(i \to j) \in \mathcal{E}} f_{i,j}(\phi_i, \phi_j).$$

Here, assuming that vehicles cross intersections always at the same default speed is sufficient to guarantee that local travel times $t_{i,j}$ and

local energy consumptions $E_{i,j}$ depend only on $\phi_i$ and $\phi_j$. In the contrary, minimizing the worst travel time over a lane would lead to a non-separable criterion:

$$t_{worst}(\vec{\phi}) = \max_{l} \sum_{(i,j)\in\mathcal{E}_l} W_{i,j} t_{i,j}(\vec{\phi}),$$

where $\mathcal{E}_l$ is the set of segments in lane $l$.

A first advantage of such separable criteria is that they can be computed efficiently in parallel, which can be beneficial for large networks. A second advantage is that each phase $\phi_i$ only influences its incoming and outgoing lane segments. As a consequence, various local search algorithms can be trivially distributed, thus dramatically reducing the complexity of the search, as detailed in the following section. But note that this separability does not prevent from falling in local optima.

## 4.2 Hill-Climbing Based Algorithm

Various local search algorithms could be considered such as hill-climbing, gradient descent (when the gradient of the evaluation function can be computed), tabu search, or simulated annealing [12]. Here we consider a simple hill-climbing approach. At each iteration $t$, for each intersection $i$, the algorithm locally searches for the best phase given the phases of neighbouring intersections $j \in N(i)$, and assigns this value to intersection $i$: $\phi_{t+1}(i)$. The details are presented in Algorithm 1, where $\Delta\phi_t$ is the radius of the search interval —which should slowly decrease to ensure convergence— and $\delta\phi_t/\Delta\phi_t$ is an integer constant.

**Algorithm 1**: Hill-Climbing-iteration($i, t, \vec{\phi}_t$)

```
1 Vmin ← Evaluate(φ_t(i), φ⃗)
2 φ_min(i) ← φ_t(i)
3 for φ_tmp ← φ_t(i) − Δφ_t to φ_t(i) + Δφ_t step δφ_t do
4     V_tmp ← Evaluate(φ_tmp, φ⃗_t)
5     if V_tmp > V_min then
6         V_min ← V_tmp
7         φ_min ← φ_tmp
8 φ_{t+1}(i) ← φ_min
```

Evaluating the performance measure at a given intersection —as done in Algorithm 1 in the function $Evaluate(\cdot)$ at lines 1 and 4— simply amounts to summing up performance measures for the 8 neighboring segments (1 segment per incoming lane, and 1 segment per outgoing lane). For each segment, vehicles do not interact with each other —because they are separated by a period $T$— so that it is straightforward to compute, as illustrated by Fig. 6:

1. when a vehicle leaving some intersection $i$ will reach the next control zone (here running at constant speed $V_{Max}$),
2. when it should go through the next intersection $j$ (here at speed $V_{Crossing}$),[4]
3. the speed profile this requires during the adaptation phase, as calculated in [15], and,
4. as a consequence, any performance measure we are interested in (traversal time, energy, ...).

A particular issue is that the evaluation function analytically computes the local criterion assuming perfect vehicles, i.e., vehicles that follow some ideal solution exactly, especially because they can change their control parameters (speed or acceleration) at any time. In some cases this may lead to phases tuned so that some vehicles do not need to slow down at all during the adaptation phase. In practice —in a simulation as in the real world— vehicles act only at discrete time steps, and thus are slightly late compared to the "perfect plan". The main consequence is that they enter a control area too late to cross it at constant speed and have to wait for the next available time interval, which is the worst possible situation. This problem is simply solved by letting the evaluation function assume that the distance between neighboring intersections is slightly longer than expected.

Finally, to prevent the optimization from ending while on an early plateaux, it is not stopped when sufficiently small (global) variations are observed, but after a fixed (hand-tuned) number of iterations — which was appropriate in our experiments.

[4] In our setting, $V_{max}$ and $V_{crossing}$ are the default speed $V$.

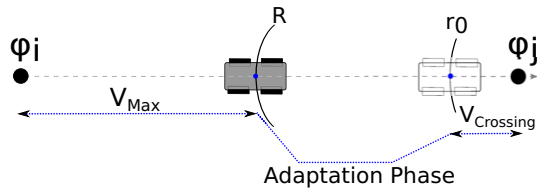


**Figure 6.** Measuring the performance for any vehicle on segment $i \to j$ requires decomposing this segment in three parts: (1) between intersection $i$ and $j$'s control zone of radius $R$, (2) the adaptation phase in the control zone, and (3) the crossing zone of radius $r_0$.

## 4.3 On-line Decentralized Version

An interesting property of the proposed algorithm is the possibility to execute it on line, with one agent $i$ at each intersection executing at each time step $t$ a local Hill-Climbing iteration and communicating its new phase $\phi_{t+1}(i)$ to its neighboors. Of course, an advantage of such an on-line optimization scheme is the ability to adapt to changing traffic conditions using only local computations. Yet, to that end the control agents need to keep on optimizing, i.e., they use constant values $\Delta\phi$ and $\delta\phi$.

**Estimating Traffic Conditions** In this on-line case, the traffic conditions are not known, but need to be estimated. Here, we employ a simple approach where each control agent $i$ estimates the flow on each of its four lanes. For lane $X$, control agent $i$ counts in $n_m$ the number of vehicles going through its intersection during minute $m$ and then estimates the flow on lane $X$ using an exponential moving average: $\bar{n}_m = (1-\alpha)\bar{n}_{m-1} + \alpha n_m$, where $\alpha \in (0,1)$ is a constant parameter that can be tuned to adapt to more or less stable conditions.

**Executing while Optimizing** One issue is that the Alternating strategy is not meant to work with continuously changing phases. First, a vehicle that has planned to go through its next intersection at time $t$ should not change its plan for time $t'$ unless it can change the remaining of its pre-computed speed profile. Second, changing the time when vehicle $v$ arrives at intersection $i$ may lead to collisions with vehicles crossing $i$ before or after $v$. This is all the more likely that the common period has been computed so as to maximize the throughput, i.e., by leaving as little free space as possible between consecutive vehicles (called the *safety distance*).

Here, the first problem is solved by not allowing to change the time when a vehicle crosses its next intersection once it has been

computed. Then, we avoid possible collisions by making a compromise between

- reducing the maximum change in phases $\Delta\phi$ (which should be large enough for the phases to adapt quickly), and
- increasing the safety distance $\epsilon$ (which should be small to minimize the period, and thus maximize the throughput).

Moreover, in Algorithm 1 the phases are not constrained to stay within $[0, 2\pi)$ to make it easier for the control agents to allocate traversal intervals to incoming vehicles (both in the Alternating and FCFS strategies).

## 5 Experiments

This section presents experiments evaluating the benefits of optimizing the phases for the Alternating strategy, and comparing this strategy to FCFS, which is expected to be more opportunistic.

All experiments are conducted on a traffic network with 6 roads and 12 intersections (Fig. 7 is a snapshot). Two groups of 6 lanes are created (one lane per road), and 4 traffic conditions are created by simply assign a high flow (15 vehicles.minute$^{-1}$) or low flow (5 vehicles.minute$^{-1}$) to all lanes in each group. These traffic conditions are noted 15–15, 15–5, 5–15, and 5–5.

Note that we measure the optimization criterion only *inside* the network (once vehicles have gone through a first intersection). This allows notably reducing the variance of our estimates —without harming the performance measure— since the system has no control over what happens outside.

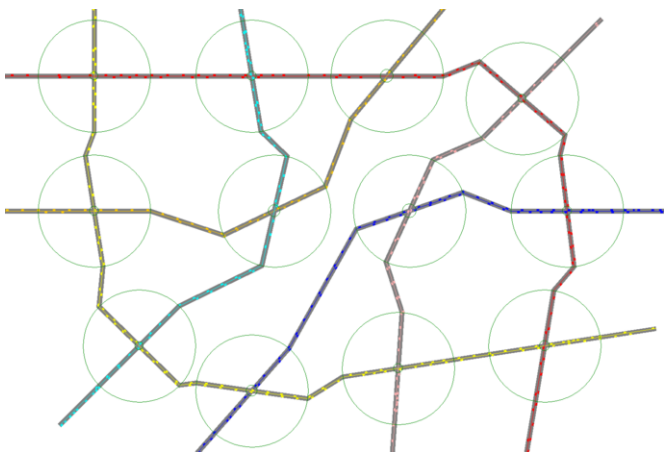

**Figure 7.** 12 Intersections Network

### 5.1 Simulation Framework

We have developed (in JAVA) a continuous-space and discrete-time simulator of a network of roads.

The detailed experimental setting for our 12-road network is the following:

- the control agents' range of action $R$ is $125m$ and $r_0$ is derived automatically as in [15];
- the distance between two adjacent intersections is $\sim 450m$ ;
- the maximum (and default) speed of each vehicle is $10m/s$, the maximum acceleration is $1.5m/s^2$ and the maximum deceleration is $-1.5m/s^2$;
- we used a near-to-near longitudinal control developed in [13] which ensures a collision-free behavior between same-lane vehicles (only outside the control zone);
- at each entrance of the network, a source generates vehicles following a Bernoulli distribution with parameter $\frac{1}{D}$, where $D$ is the average time, in seconds, between two consecutive injections (For example $D = 4$ implies an average of 15 vehicles per minute);
- the vehicles are $5m$ long and $2.5m$ wide, and the safety margin of vehicle inter-distance $\epsilon = 1.5m$;
- the simulation time step is set to $0.05s$.

In this case, the common (i.e., worst) period in the network of Fig. 7 is $T_{max} = 3.5s$. We fixed the parameters of the hill climbing optimization, $\Delta\phi = 0.05s$ and $\delta\phi = \frac{\Delta\phi}{10}$.

A video showing the simulator can be viewed at `http://www.loria.fr/%7emtlig/videos/12_intersections.avi`.

### 5.2 Off-Line Optimization

We first consider the off-line case, comparing Alt and FCFS either (asynchronous case) with local periods $T_{min}(i)$ and random phases, (synchronous case) with common period $T$ and random phases, or ("optimized" case) with the common period $T$ and phases optimized for Alt. Each of the 6 resulting strategies is used on the 4 traffic conditions during 10 experiments lasting 1 hour each (simulation time). The results are presented in Table 1, where we measure the average consumed energy per vehicle (considering only the kinetic energy used to accelerate).

**Table 1.** Average energy consumed per vehicle using Alt and FCFS in either their asynchronous, synchronous or optimized versions (see text); Injections are measured in vehicles/minute

| Injections | | $(10k\ veh)$ 15–15 | $(7k\ veh)$ 15–5 | $(7k\ veh)$ 5–15 | $(3.5k\ veh)$ 5–5 | Ave. |
|---|---|---|---|---|---|---|
| FCFS | async | 137.6 ±2.4 | 104.6 ±2.1 | 102.3 ±2.2 | **60.3** ±0.6 | 101.2 |
| | sync | 178.8 ±4.6 | 130.7 ±5.7 | 128.6 ±5.7 | 67.4 ±3.2 | 126.4 |
| | optim | 176.9 ±2.5 | 131.6 ±4.7 | 129.8 ±3.5 | 66.1 ±4.1 | 126.1 |
| Alt | async | 111.1 ±1.3 | 107.1 ±0.8 | 102 ±0.8 | 85.9 ±0.8 | 101.5 |
| | sync | 92.6 ±6.4 | 87.7 ±10.2 | 88.6 ±8.6 | 88.5 ±5.1 | 89.4 |
| | optim | **83.8** ±7.4 | **80.2** ±7.4 | **74.7** ±5.6 | 83.7 ±12.4 | **80.6** |

A first observation is that synchronizing the network alone already improves the traffic flow for Alt, except in low traffic conditions, while it always degrades the traffic flow for FCFS (which benefits from being opportunistic in the asynchronous case).

Then, for synchronous intersections, using optimized phases allows Alt to reduce the energy consumption –meaning that "green waves" are efficiently created– while this has no impact on FCFS –i.e., these phases are not optimal for FCFS.

Also, as FCFS is opportunistic, it saves energy in low density traffic conditions (5–5). Yet, the higher the density of traffic conditions, the more conflicts happen at intersections (i.e., the less opportunities can be exploited by FCFS) meaning that vehicles often have to slow down. In the same high-density traffic conditions, Alt benefits from green waves and maintains a low energy consumption.

Note that measures with the travel times criterion lead to the same conclusions because, in both cases, the criteria try to keep the vehicles at their default speed for as long as possible.

### 5.3 On-Line Optimization

We now consider a 4-hour period during which the four traffic conditions used previously are applied 1 hour each, and compare:

- FCFS with local periods and random phases,
- Alt with phases optimized off-line using the average flow over 4 hours: 10–10, and
- Alt with phases optimized on-line using 4 learning rates $\alpha$.

**Table 2.** Average energy consumed per vehicle using Alt with off-line optimized phases, Alt with on-line optimized phases, and FCFS with random phases

| Alg. | Alt on-line | | | | Alt off-line | FCFS |
|---|---|---|---|---|---|---|
| ($\alpha$) | (0.05) | (0.1) | (0.2) | (0.3) | | |
| Energy | 78.2 ±4.3 | 77.1 ±4.7 | **76.1** ±5 | 77.1 ±4.4 | 80.1 ±6.9 | 101.2 |

As detailed previously, each agent intersection $i$ estimates alone the flows on its lanes; communicates its phase only with its neighbors; and optimizes its own phase $\phi(i)$ using Algorithm 1. Table 2 presents empirical estimates of the energy consumption per vehicle in each case. The on-line version of the Alt strategy outperforms both its off-line version and (by a wider margin in this case) FCFS. This clearly demonstrates that the on-line estimation of the flows and optimization of the phases are efficient, leading to an adaptive traffic network. The best results are obtained with $\alpha = 0.2$, though the difference with other values does not seem significant.

## 6 Discussion

**Dealing with More Complex Scenarii** In this work, the Alternating strategy has two main limitations: (1) at a given intersection, all lanes use the same amount of time, while the flows may be very different from one lane to the next, and (2) the vehicles cannot turn.

For the first limitation, one may simply use an $n/m$ alternation pattern ($n$ time slots for one road, alternating with $m$ time slots for the other road) at each intersection rather than 1/1.

As in most traffic light approaches (see Sec. 2), the second limitation can be overcome by using, at each intersection, cycles made of 4 periods instead of two: 2 periods to let cars from each road go straight or turn right, plus 2 periods to let cars from each road turn left. Lanes need to be duplicated before the intersections To prevent conflicts between cars from the same lane but going in different directions.

In both cases more work is needed to compute the various durations in the periodic signal, and to optimize the phases (using more complex analytical evaluation formulas).

**Going Further with Alt+FCFS** As observed, applying phases optimized for Alt while using FCFS leads to degraded results. So, one interesting question is whether phases could be optimized specifically for FCFS. Because FCFS is opportunistic, when a vehicle leaving intersection $i$ will go through the next intersection $j$ depends on potential conflicts with other vehicles, and is thus difficult to predict. As a consequence, it is more difficult to analytically compute a performance criterion on segment $i \rightarrow j$ given $\phi(i)$ and $\phi(j)$. This implies (1) that simulations may be necessary to evaluate a given set of phases $\vec{\phi}$, and (2) the optimization cannot be distributed because of complex interactions between intersections.

Another interesting problem is how to make the best of both FCFS and Alt. A first idea would be to switch between them depending on the overall traffic conditions. But one could even envision heterogeneous traffic networks in which some of the intersections use an Alt controller (dense traffic), and other rely on FCFS (low traffic).

## 7 Conclusion

In this paper we have first proposed an approach to optimize the flow of a traffic network in which vehicles do not need to stop at intersections. This approach significantly improves on simply using independent stop-free intersections. It is also notably better than a more opportunistic First-Come First-Served strategy —which prevents the formation of green waves— under dense traffic conditions. Moreover, the algorithm can be easily adapted to work on line, each intersection continuously estimating the local traffic conditions and optimizing its phase —only requiring to communicate this phase to its neighbors. This on-line version significantly improves on the off-line one, showing its adaptability to changing flows.

Further experiments should be conducted to get a better understanding of the approach, in particular considering different networks and different scenarios. Also, some parts of the approach could be improved such as the local-search algorithm —e.g., using the Distributed Stochastic Algorithm, a variant of Hill-Climbing especially designed for such distributed settings [16, 14]— or the estimation of the traffic conditions.

## REFERENCES


[1] L.C. Bazzan, 'Opportunities for multiagent systems and multiagent reinforcement learning in traffic control', *Autonomous Agents and Multi-Agent Systems*, **18**(3), 342–375, (2009).

[2] D. De Oliveira and L.C. Bazzan, 'Traffic lights control with adaptive group formation based on swarm intelligence', in *Ant Colony Optimization and Swarm Intelligence*, 520–521, Springer, (2006).

[3] K. Dresner and P. Stone, 'Multiagent traffic management: A reservation-based intersection control mechanism', in *Proc. of AAMAS*, (2004).

[4] K. Dresner and P. Stone, 'Multiagent traffic management: An improved intersection control mechanism', in *Proc. of AAMAS*, (2005).

[5] C.L. Fok, M. Hanna, S. Gee, T.C. Au, P. Stone, C. Julien, and S. Vishwanath, 'A platform for evaluating autonomous intersection management policies', in *Third International Conference on Cyber-Physical Systems (ICCPS)*, pp. 87–96. IEEE, (2012).

[6] J-J. Henry, J-L. Farges, and J. Tuffal, 'The prodyn real time traffic algorithm', in *IFACIFIPIFORS CONFERENCE ON CONTROL IN*, (1984).

[7] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and R. I. Winton, 'SCOOT - a traffic responsive method of coordinating signals', Technical report, TRRL, (1981).

[8] PR Lowrie, 'The sydney coordinated adaptive traffic system-principles, methodology, algorithms', in *International Conference on Road Traffic Signalling*, (1982).

[9] R. Naumann and R. Rasche, *Intersection collision avoidance by means of decentralized security and communication management of autonomous vehicles*, Univ.-GH, SFB 376, 1997.

[10] R. Naumann, R. Rasche, J. Tacken, and C. Tahedi, 'Validation and simulation of a decentralized intersection collision avoidance algorithm', in *Intelligent Transportation System (ITSC), IEEE*, (1997).

[11] D.I. Robertson, 'Transyt: a traffic network study tool', (1969).

[12] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, NJ: prentice Hall, 1995.

[13] A. Scheuer, O. Simonin, and F. Charpillet, 'Safe longitudinal platoons of vehicles without communication', in *Proceedings of the international conference on Robotics and Automation (ICRA)*, (2009).

[14] M. Smith and R. Mailler, 'Improving the efficiency of the distributed stochastic algorithm (extended abstract)', in *Proc. of AAMAS*, (2010).

[15] M. Tlig, O. Buffet, and O. Simonin, 'Decentralized traffic management: A synchronization-based intersection control', in *Proc. of the International Conference on Advanced Logistics and Transport*, (2014).

[16] Z. Weixiong, W. Guandong, X. Zhao, and W. Lars, 'Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks', *Artificial Intelligence*, **161**(1–2), 55–87, (2005).

[17] M. Wiering, 'Multi-agent reinforcement learning for traffic light control', in *ICML*, pp. 1151–1158, (2000).