The KISS Principle Applied to Traffic Light Behaviours

Philippe Mathieu, Antoine Nongaillard, Alexandre Thery

The KISS Principle Applied to Traffic Light Behaviours

Philippe Mathieu[1][0000−0003−2786−1209], Antoine Nongaillard[1][0000−0001−8551−0509], and Alexandre Thery[1]

Univ. Lille, CNRS, Centrale Lille, UMR 9189 – CRIStAL (équipe SMAC), F-59000 Lille, France `firstname.name@univ-lille.fr`
http://cristal.univ-lille.fr/en/gt/i2c/

1 Introduction

There have been numbers of papers trying to find an optimal solution for the traffic regulation problem by implementing intelligent behaviours into traffic lights. The goal is to obtain an intelligent infrastructure which allows the different flow of cars to pass one or multiple intersections as efficiently as possible in terms of waiting time, to avoid congestion and annoyance among drivers, which can lead to road accidents.

Many have tried to develop such a behaviour in the past decades, using various approaches, but today machine learning is mostly used. We believe that the resulting behaviours, while having great performances theoretically, are not applicable in real life as long as humans continue to drive cars themselves, and that the complexity of this approach is at cause.

In this paper we will show that it is possible to design a descriptive behaviour with performances similar to what allows machine learning, using a more sober approach focused on the essential elements ([3], [1]). This behaviour can thus easily be deployed on any crossroads, and is more maintainable, upgradable and legally acceptable. This will be illustrated on an area of the city of Nice.

Fig. 1. The area of Nice on which our work is set, on OpenStreetMap (left) and in the SUMO simulator (right).

2 Intelligent traffic lights: state of play

Hereafter is a list of the main approaches used by researchers during the past design traffic light behaviours in order to solve the traffic regulation problem.

Equation-based approaches try to formalize the problem in the form of equations, usually concerning the flow of vehicle and the different rules of the infrastructure they travel on. The different equations are then computed in order to find an optimal solution for the problem. Strong hypothesis are required in order to create an equation system simple enough to be computable, and he modeled magnitudes are purely mathematical: there is no notion of individual vehicle, so it is impossible to obtain results on a scale more precise than global (like the maximum waiting time of a vehicle at an intersection). [5] is a good example of this problem.

Multi-agent simulations are capable of providing adequate tools to work on the traffic regulation problem. They are able to overcome the limitations of equation-based approaches. Most importantly, multi-agent simulations focus on the essence of the problem: the behaviours, which can appear on vehicles, vehicle generators, vehicle wells, and traffic light, and which are descriptive and thus explainable.

Note that **data-based approaches** are - most of the time - a particular case of simulation-based approaches, since they often rely on a simulation to generate car flows from the data and to provide an accurate reward function to the machine-learning algorithm. Simulations are also used once the behaviour has been generated, to test it.

Data-based approaches, like machine-learning, allow to obtain behaviours capable of following the general evolution of the traffic relatively to various periods of time: a day (rush hour / night), a week (work days / weekends), a year (holidays / work period). They use data usually concerning entry and exit flux of a portion of infrastructure (a crossroads for instance), acquired by placing sensors on the road that count the vehicles. These behaviours are unable to follow small punctual perturbations, like road accident, or traffic caused by exceptional events, because these event are drowned in the mass of "usual" data. In some way we could say that these behaviours correspond to the average traffic configurations: as you will see later, our approach try to show that there is no necessity to use machine learning to obtain such a behaviour.

The behaviours resulting from a machine-learning approach are not explicable and consequently cannot be maintained: if a problem linked to the behaviour of the traffic light occurs, a vast part of the study has to be redone.

Our approach, in order to have the same features that one obtained via machine-learning, require the same type of data. As a consequence, our behaviour shares some of its flaws that are directly linked to the usage of data ; but we resolve the problem of the explainability, which is a major brake to the implementation, mainly because of the jurisdiction linked to road safety.

3 General notions for our approach [1]

Among the road traffic simulators in existence, we have chosen to work with SUMO. Its microscopic approach and its focus on the traffic scale seems to correspond perfectly to our needs, whereas other simulators like SCANNeR and VTD, for example, have a nanoscopic approach more centered on a specific car and the traffic around it.

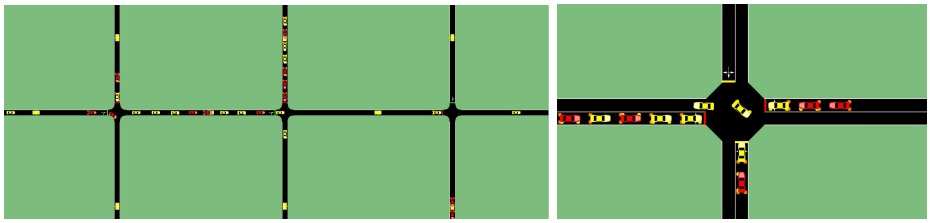


Fig. 2. Our evaluations are tested on a Manhattan-style grid infrastructure. On the left you can see three of our nine crossroads regulated by traffic lights with their sensors. On the right, a zoom on one of the crossroads with some vehicles already engaged.

Within SUMO, our work uses a classical abstract model [2,6] but also GIS real maps for testing. Our abstract model is parametric and consists of a grid of n verticals $\times$ n horizontal one-way lanes, with a generator at one end (left and top of the grid), and 1 pit at the other end (bottom and right). This implies $2n$ generators and as many wells.

Each intersection is regulated by a traffic light (making n^2 lights to be managed), knowing that in everything that follows what we call a "traffic light" is actually the group of lights of a given intersection, which in our case corresponds to two real life traffic lights: one for each incoming lane. It is necessary, in our opinion, to connect multiple traffic light in order to verify that their interactions are not destructive, hence our $n \times n$ grid. Each traffic light has two sensors, one on each incoming lane, that can detect the passage of a vehicle and inform the light (they are represented in SUMO by yellow rectangles on the lanes: see Fig.2).

The generators send their vehicles at one of the $2n$ pits randomly. We created 2 kinds of generators: one with a high-frequency spawn rate, and the other a low-frequency one. Each vehicle generated also has one chance out of two to have the ability to drive over the speed limit of the lane (and it appears red in the simulation instead of yellow): this is useful for some behaviours we have tested, as you will see.

We run the simulation for every possible generator combination, which allows $2^{(2n)}$ configuration, constituting an exhaustive set. With $n = 3$ this leads to 6

[1] illustrated in the video on the Youtube SMAC Channel

generators, 9 lights and 64 possible configurations, which allows us to compare objectively.

This approach can easily be used on real maps (Fig.3), to test a behaviour on a grid that fits more accurately the reality. But it is our opinion that this should only serve as demonstration or validation, and not to design behaviours, since it is impossible to obtain exact real life traffic conditions, and even if it was the designed behaviour would only fit these particular conditions and its overall performances would remain uncertain.

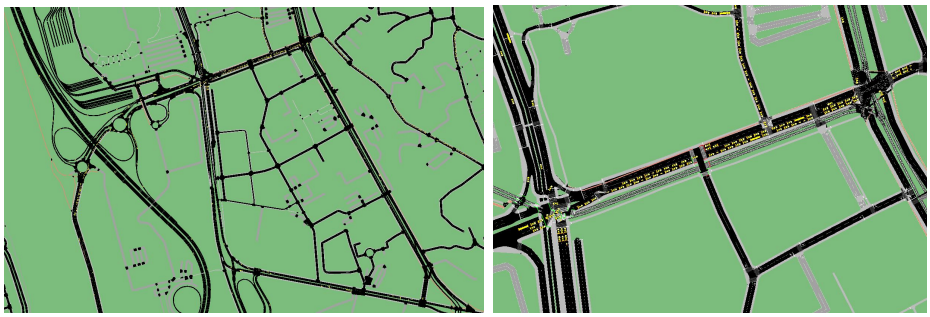


Fig. 3. On the left a simulation of an area of the city of Nice. On the right a zoom on crowded intersections. You can see the different vehicles blocked at the traffic lights.

We have tested five traffic light behaviours named `timer`, `timer phased`, `count`, `punish` and `reward` on our grid. It is important to note that for a given traffic light, the lights on each incoming lane are linked: when one is green, the other is red, and vice-versa. When a traffic light changes its state, it passes briefly by a state where the previously green light turn to amber, to indicate to the incoming cars that they need to brake, before turning to red. The behaviours are described below:

- **Timer** is the current behaviour of most of the traffic lights around the world. All the traffic lights are synchronized and change their state once a certain time has passed.
- **Timer phased** is the same as timer, but one out of two traffic lights are phased: when one is letting the horizontal lane pass, its neighbors are letting the vertical lane pass and vice-versa.
- **Count** is an adaptive behaviour. It counts the number of vehicle on both incoming lanes every given period and turns green for the lane where they are the most. If there is the same number of vehicles on both lanes, it does not change its state.
- **Timer%** is a mix of `timer` and `count`: it has the same period of operation than `timer` but the portion of time it is green for a certain lane is proportional to the number of cars on it. In other words, it has a finite period of operations with green phase's duration calculated according to the density of vehicles on each lane.

– **Punish** uses the sensors that we have put on the incoming lanes: if a vehicle driving over the speed limit is detected, the light turns red for its lane.
– **Reward** is similar to punish: it turns green for the lane where vehicles driving below the speed limit are detected.

`Punish` and `Reward` are two intelligent behaviours that are currently tested by various towns across the world. They are not relevant for our problematic since they do not aim at reducing traffic jams or waiting time but speeding, but are to be mentioned.

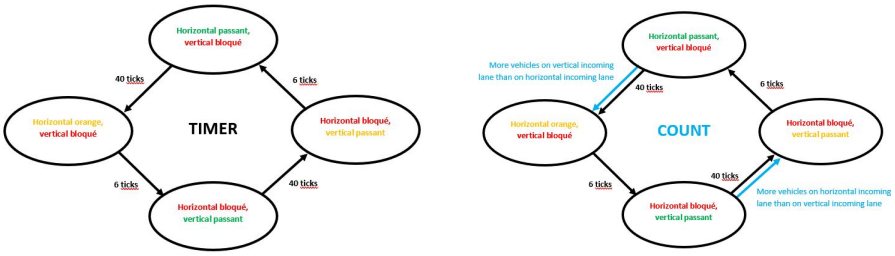


Fig. 4. State diagrams of the different traffic light behaviours

Some natural constraints on traffic light behaviours are stated below and need to be taken in account in order to obtain a realistically implementable behaviour:

– A red light does not only stop vehicles, it also let pedestrians cross the road. It is not possible, in a general case, to use behaviours that can stay in a given state for a very long time, even more so according to the position the traffic light (crowded town centers for example)
– Moreover, for this reason and because of the drivers reaction time, the light must stay in a given state for a minimum time.

Consequently, a traffic light necessarily has its state bounded between a maximum and a minimum value.

4 Traffic configurations

In order to find a behaviour that can work with any situation, we have to test our behaviours in different traffic configurations. It is possible to discretize the different traffic configurations by separating them in tree distinct categories: fluid balanced traffic, dense balanced traffic, and unbalanced traffic.

We have tested our behaviours on these different traffic configurations, to be able to tell which one has the best results for every given situation. In order to evaluate their performances as precisely as possible, we have used some of the

metrics introduced in [4] and have added some of our own thanks to the data provided by SUMO and which could be relevant for city planners, like carbon emissions and fuel consumption.

4.1 Fluid balanced traffic

The traffic is light and similar for every incoming lane: there is no waiting queue and few cars. Hereafter are the results for an exhaustive set of traffic configurations where the traffic stays light (low frequency generators spawn a car every 20 ticks, and high-frequency ones one every 10 ticks):

Behaviour	**Timer**	**Timer phased**	**Count**	**Timer%**	**Punish**	**Reward**
Co2 emissions (g)	232.82	282.44	**214.12**	261.71	291.09	298.67
Fuel consumption (ml)	100.08	121.41	**92.04**	112.50	125.13	128.39
Waiting time (s)	27.91	45.01	**14.47**	36.29	47.57	52.07
Speeding time (s)	3.88	2.83	3.12	3.34	3.30	3.37
Speeding time (%)	7.28	4.08	5.69	5.70	4.25	4.75

Table 1. Results for light traffic: 103,680 cars simulated through the exhaustive sets for each behaviour

The `count` behaviour has the best overall wait time, and the lowest fuel consumption and CO2 emissions. Even if waiting times are low in all cases (little traffic), `count` surpasses all the other behaviours by far. This result is only logical: `count` will always be best in light traffic situations because its ability to adapt to the vehicles that are coming: it can let the pass as it sees them coming without making them wait most of the time.

4.2 Dense balanced traffic

The traffic is dense and similar for every incoming lane: a lot of car are waiting in queues on every lane. To represent this we have set our simulation as follows: low frequency generators spawn a car every 6 ticks, and high-frequency ones one every 3 ticks. The results are the following:

The `timer` behaviour has the best overall wait time, the lowest fuel consumption and CO2 emissions. The reason why `count` is inefficient here is that the portion of the lanes where it counts vehicles are always full: it end up switching its state repetitively as soon as a few vehicles have crossed and its measuring period has passed (which is far shorter than the period of `timer` for obvious reasons). We fall on the natural conclusion that in dense traffic conditions, the `timer` behaviour is to be privileged, because there is no real decisions to make when a crossroad is uniformly jammed.

4.3 unbalanced traffic

These cases are the most tricky and correspond to most of the situations encountered in real life. They regroup all the traffic configurations where one or more lanes are flooded by cars (dense) while at least one other isn't (fluid). To represent this we have set our simulation as follows: low frequency generators spawn a car every 20 ticks, and high-frequency ones one every 5 ticks. The results are the following:

The `timer%` behaviour has the best overall wait time, the lowest fuel consumption and CO2 emissions. This is an intuitive result: the traffic configuration is a mix of dense and fluid traffic, and consequently the most adequate behaviour is a mix of `time` and `count`.

5 Our proposition: slot-based meta behaviour

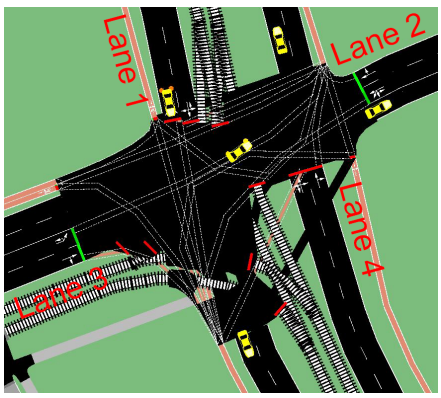
By synthesizing the optimal behaviours we got for every type of traffic configuration we are able to approach the type of performances that a behaviour obtained using a machine-learning algorithm would give, as described in part 2.3, using the same type of data in a simpler manner. We use the following method for the crossroads where the intelligent traffic light is to be implemented:

- Using the flow data for each incoming lane, we calculate the average density of vehicles for each given time slot (15min for example). The smaller the time slot the smaller the granularity of the behaviour will be: it will react to changes in traffic density more quickly, but will be more prompt to untimely changes of state.
- We form the different periods of operation of the traffic light by regrouping the average densities of every lane for each time slot and determining to which traffic configuration it corresponds: fluid, unbalanced or dense.
- We attribute every period of operation its optimal behaviour according to its traffic configuration(see 10.1).

We obtain a meta behaviour that is able to switch between optimal behaviours according to the traffic configuration every time slot: we call this behaviour `slot-based meta behaviour`.

Note that the `slot-based meta behaviour` can, in very much the same way as a reinforcement-learning algorithm would, use sensors to continue to acquire data and thus improving its capacity to follow the changes in the traffic. However we also find the same liability in its incapacity to take in account punctual variations in the traffic, like accidents or exceptional events.

We have tested our `slot-based meta behaviour` on a crossroads of the city of Nice and have calibrated the flow of vehicles using data from the TomTom map API (Fig.5). We obtain a flow of vehicles with a density that follows to real-life variations: between 8pm-7am there are practically no cars in circulation, and between 7am-9am and 5pm-8pm there are rushes on work days. We have chosen to work on a Monday to limit the simulation time.



	Sun	Mon	Tue	Wed	Thu	Fri	Sat
12:00 AM	7%	2%	2%	3%	4%	5%	6%
	3%	0%	0%	0%	0%	2%	3%
02:00 AM	1%	0%	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%	0%	0%
04:00 AM	0%	0%	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%	1%	0%
06:00 AM	1%	5%	5%	5%	5%	5%	1%
	3%	37%	41%	35%	38%	35%	6%
08:00 AM	6%	51%	60%	44%	57%	51%	11%
	11%	34%	41%	32%	39%	37%	19%
10:00 AM	17%	30%	35%	33%	33%	35%	27%
	23%	31%	34%	34%	34%	37%	31%
12:00 PM	22%	27%	28%	34%	30%	33%	29%
	15%	24%	25%	28%	26%	26%	22%
02:00 PM	17%	28%	30%	34%	31%	33%	23%
	18%	32%	34%	36%	35%	39%	27%
04:00 PM	21%	44%	48%	46%	50%	58%	28%
	26%	59%	69%	61%	70%	75%	29%
06:00 PM	26%	45%	56%	53%	58%	63%	30%
	21%	25%	31%	31%	34%	40%	28%
08:00 PM	13%	12%	14%	15%	16%	19%	18%
	8%	8%	9%	9%	10%	10%	11%
10:00 PM	6%	7%	8%	8%	11%	9%	10%
	5%	5%	6%	8%	11%	10%	10%

Fig. 5. The crossroads where the implementation of an intelligent traffic light will be tested in SUMO (left) and the data from the TomTom API we used to calibrate the vehicle flows trough Nice (right).

The test we have conducted is simple: first, we have tested the `timer` behaviour on our crossroads, with the traffic configuration described above. Additionally to the gathering of the results of the `timer` behaviour, we have measured the density of vehicles for each entry lane of the crossroads. We have then used this measured density to determinate the time slots and the periods of operations for our `slot-based meta behaviour`. We have found that the traffic becomes unbalanced for the two daily rushes (approximately 7am-9am and 5pm-8pm) and is fluid the rest of the time. Finally, we have tested our `slot-based meta behaviour`, with these periods of operations, and with the same traffic configuration. The results are shown in Fig.6.

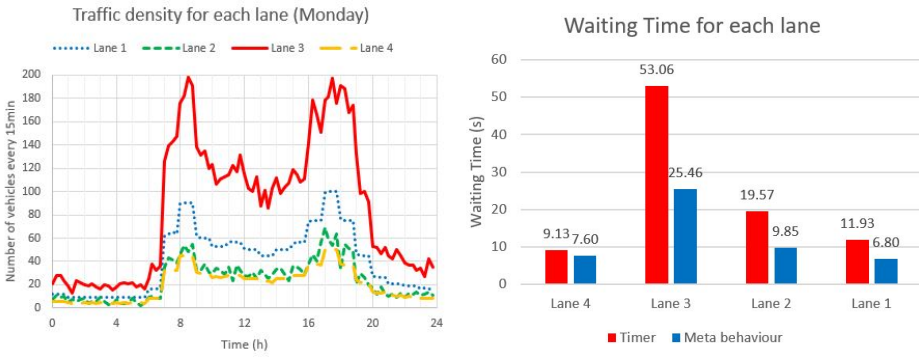


Fig. 6. Some results from our experiment on a crossroads of the city of Nice: the density of vehicles for each lane (left) and their average waiting time for the classic timer behaviour and our slot-based meta behaviour (right).

We see that our `slot-based meta behaviour` has divided the waiting time of lanes 3 and 2 by half compared to the `timer` behaviour. For the lanes 1 and 4 the time saving is less pronounced, because it is already low, making it more difficult to reduce.

6 Conclusion

We believe that the question of the explainability is a major brake to the implementation of intelligent traffic lights, whatever the performances, the same way it currently is a major brake for the deployment of autonomous vehicles. For this reason, among others, until this day only a few intelligent traffic lights exist on the field: we can cite `punish` and `reward` behaviours, which aim at reducing speeding and are currently being tested near critical areas like schools. Their success is simple to explain: they have a simple, local objective, do not need any massive change in the infrastructure to function and their behaviour is fully explainable.

In these few pages we have shown that by aggregating the optimal simple behaviours obtained for every traffic configuration we are able to design a behaviour capable of following the variations in the traffic from the data, like an behaviour obtained via machine-learning would, but that is fully explainable, and so easier to implement, maintain and upgrade. We hope that our work has convinced you that there is not so much the need of great complexity to work on traffic regulation problems on the scale of one or multiple successive crossroads, if you want to obtain a behaviour that is realistically implementable, and that more sober approaches, that focus on the essential elements of the problem should be privileged accordingly to the KISS/Occam's razor principle ([1], [3]).

References

1. Kiss principle. https://en.wikipedia.org/wiki/KISS_principle.
2. Manhattan mobility model. https://en.wikipedia.org/wiki/Manhattan_mobility_model.
3. Occam's razor. https://en.wikipedia.org/wiki/Occam%27s_razor.
4. Kurt Dresner and Peter Stone. Multiagent traffic management: A reservation-based intersection control mechanism. In *Proceedings of the Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'04), Volume 2*, pages 530–537, 2004.
5. Xiaojian Hu, Jian Lu, Wei Wang, and Ye Zhirui. Traffic signal synchronization in the saturated high-density grid road network. *Computational intelligence and neuroscience*, 2015, 2015.
6. Geetha Jayakumar and G Gopinath. Performance comparison of manet protocols based on manhattan grid mobility model. *Journal of Mobile communication*, 2(1):18–26, 2008.