EPiC Series in Computing

Volume 62, 2019, Pages 244–255

SUMO User Conference 2019

Remarks on Traffic Signal Coordination

Peter Wagner[1,2,*], Robert Alms[3,†], Jakob Erdmann[4,‡] and Yun-Pang Flötteröd[5,§]

[1] Deutsches Zentrum für Luft- und Raumfahrt,
Institute for Transportation Systems, Berlin `peter.wagner@dlr.de`
[2] Technical University of Berlin,
Institute for Land- and Sea Traffic, Berlin `p.wagner@campus.tu-berlin.de`
[3] Deutsches Zentrum für Luft- und Raumfahrt,
Institute for Transportation Systems, Berlin `robert.alms@dlr.de`
[4] Deutsches Zentrum für Luft- und Raumfahrt,
Institute for Transportation Systems, Berlin `jakob.erdmann@dlr.de`
[5] Deutsches Zentrum für Luft- und Raumfahrt,
Institute for Transportation Systems, Berlin `yun-pang.floetteroed@dlr.de`

Abstract

The co-ordination between traffic signals is assumed to be important for the good organization of a transport system. By using an artificial approach to create and analyze a multitude of transportation systems, a few different simple traffic signals programs has been put to the test and compared to each other. The result is that a well co-ordinated system can be outperformed by a non-coordinated signal set-up, where all signals controlers run in (single intersection) actuated mode. Clearly, these results are preliminary and require more investigation.

1 Introduction

Good, or even optimum traffic signal settings in real nets are difficult to obtain, and it is often not clear, to which kind of vehicles they apply. For instance, when designing a green wave, the mathematical formulation behind assumes highly idealized vehicles that have infinite acceleration capabilities, and do not show any platoon dispersion – the bands of the green wave are assumed to stay sharp forever. This idealization is also behind some mathematical formulation when trying to optimize the signal setting in a whole network. It is clear that this is a useful approximation, but as all approximations, it has its limitations.

It is quite another story what is done in real life. Anecdotal evidence seems to imply, that real road networks are often organized in a quite mixed fashion, where most signals are locally optimized to a certain degree, together with a few green waves where in addition traffic is co-ordinated at least along arterials that bear a strong traffic demand.

*Wrote most of the text, simulation of the synthetic nets
†Simulation and description of Berlin Center
‡SUMO specialist, helped with the simulation, wrote TLSCoordinator.py
§Wrote and adapted tlsCycleAdaptation.py

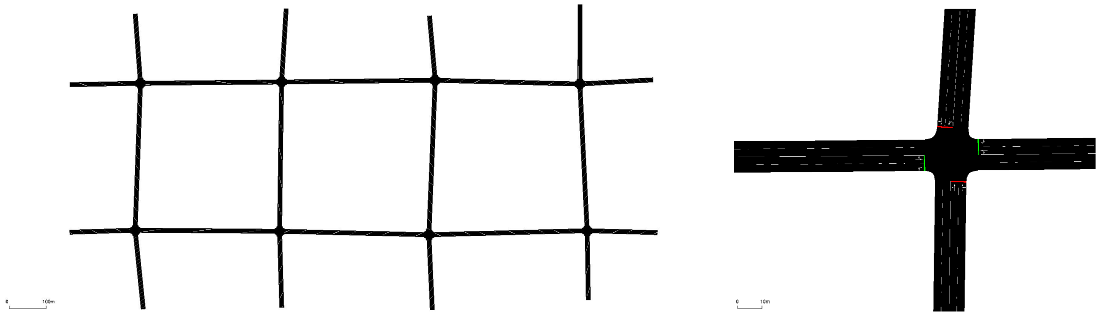
Figure 1: Screenshot of one 4x2 network and a zoom into one of the intersections.

While arterial co-ordination can be demonstrated to yield gains in efficiency under fairly mild conditions (see, e.g. the supplemental material in the appendix C for an example), the co-ordination of a whole transport system is not as simple. In addition to the mathematical and organisational challenges that come with this task, it is also not clear what can be gained. So, an optimum solution might turn out to be just a few percent or so better in reducing delays, emissions, and even crashes, leading to the question whether it is worth the effort.

A considerable number of approaches to design and optimize traffic signal control in networks has been described in the literature. While, to the knowledge of the authors, a good overview article is missing, a certain restricted one may be gotten from [6].

2 Methodology

To better understand what co-ordination can and what it cannot do, the following approach is used. All the work is being done with synthetic grid networks, typically grid networks with 8 to 112 intersections. Little effect of the system size has been spotted, but this has not systematically looked for. The links between the intersections were chosen to be roughly 400 m long, and each link was bi-directional with two lanes per direction. This has been chosen intentionally to enable platoon dispersion by having vehicles with different preferred speed and therefore overtaking is possible. Slight modifications have been introduced by disturbing the position of each intersection from the ideal grid, so that each network is unique, see Figure 1 for one realization.

Such a network is then driven by a random demand. It has an overall component (its level) and a random component: running the procedure to generate a random demand twice does not yield the same trips, even if the level of demand has not changed. The following approach generates for each network five different levels of demand (from small to large), and for each level five different realizations. These trips are used as they are, no computation of a user equilibrium is undertaken.

Each network with each single demand is then used to run four simulations:

fix A run with a default system of traffic signals, which is generated by SUMO's `netgenerate` with the consideration of the number of lanes on roads and protected left-turn phases. More details about SUMO's automatically generated TLS-Programs can be found in the Wiki [1].

fixSC Optimizing each individual intersection to the parameters put forward by Webster's approach. [5]; a bit contrary to conventional wisdom, this creates intersections with different cycle times.

fixSCO The optimized intersection settings from the second approach were searched for the maximum cycle time, and then all traffic signals were set to this cycle time. On top, a simple co-ordination approach tried to change the offsets of the signals to improve the co-ordination between the signals.

actd All the traffic lights are switched to actuated ones. In almost all simulation runs that have been conducted this turned out to be the optimum choice. Note, that actuation does by no means guarantee co-ordination, an actuated controler tries to optimize a single intersection.

There are six tools that have been used to run this research program, they all are part of the SUMO suite [4]: the programs `netgenerate`, `netconvert`, and `sumo` to generate and transform (for different traffic signal set-ups) the networks and run the microscopic simulation, and the python scripts `randomTrips.py`, `tlsCycleAdaptation.py` [2], and `tlsCoordinator.py` to generate the demand and to compute good traffic signal settings. The latter two are described in appendix B.

3 The results

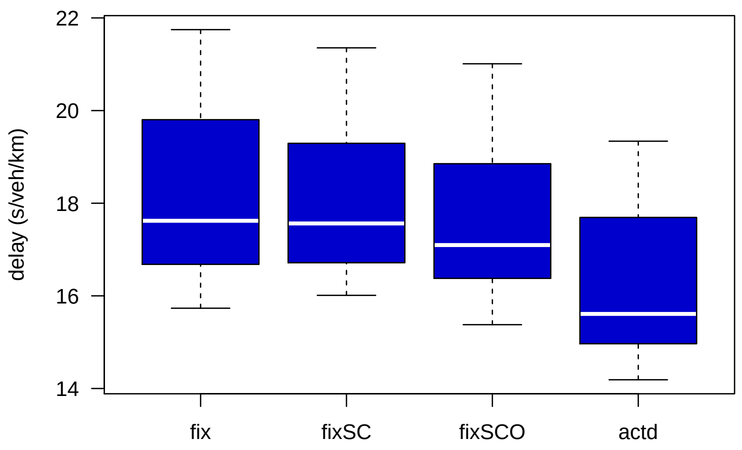

Figure 2: Box-Whisker plot of the delays in the networks for the four methods.

The following simulations used four different network sizes (4×2, 6×4, 10×6, 14×8), six different levels of demand ($p = (16, 8, 4, 2, 1, 0.5)$), leading to flows between 225 ... 4800 veh/h, and the four methods described above. Each network/demand combination was repeated five times, where each repetition used the same network, but a different demand, i.e. a different set of trips by a fresh call to `randomTrips.py`. In the simplest version, all the delays of all the simulated vehicles have been collected into a mean-value, and all these mean-values have been stored for further analysis, resulting in 120 different mean delay values. To account for the different size of the networks, each delay is normalized to the route-length, so the metric is delay per vehicle per kilometer. They are compared in Figure 2.

In these scenarios, organizing intersections according to Webster's theory gives already a strong improvement compared with SUMO's default. Very weak additional gains can be gotten by switching on the co-ordination.

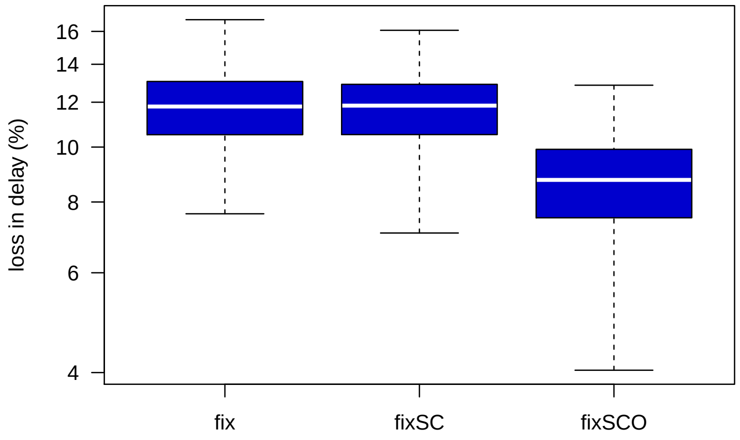

Figure 3: Box-Whisker plot of the relative difference in delays in the networks compared to the actuated control.

However, SUMO's actuated controller still improves on this despite the fact that actuation does not act in a co-ordinated manner. All in all, for the simulations done here, the median values for the four methods are $d = 17.6, 17.6, 17.1, 15.6)$ $s/veh/km$, i.e. the actuated controller beats the co-ordinated control by about 10%, see Figure 3. The numbers that constitute this boxplot are the differences between each average delay of one run and the delay produced by the actuated control for the same run (specified by the same network and demand), i.e. $d_i^{\mathrm{X}} - d_i^{\mathrm{actd}}$ where d_i^{X} is the delay produced by method X={fix,fixSC,fixSCO} in run i. A look at the numbers confirms, that the actuated control was always better, and it could be seen that the co-ordinated fixed time setting (fixSCO) give a small improvement (10.7% worse than actd) over the fixed time setting (fixSC) without co-ordination (12% worse than actd), see Figure 3.

3.1 The Berlin scenario

To put the results above into the context of real networks, a similar simulation of the inner city of Berlin has been undertaken. Here, a realistic time-dependent demand has been used, a realistic network has been used, too, but it was not possible to get the real traffic signal settings for this network. So, it made a fine test-case for the four methods described in section 2.

The network is displayed in Fig. 4; it contains 120 signalized intersections, had a total length of 241.78 km, and during the 24 hour simulation, about 190,000 vehicles crossed the network. The number depends a bit on the traffic signal settings since the network is, especially during the rush-hours, at the border of capacity. This leads in some of the scenarios to a lot of teleports, if some of the intersections get grid-locked.

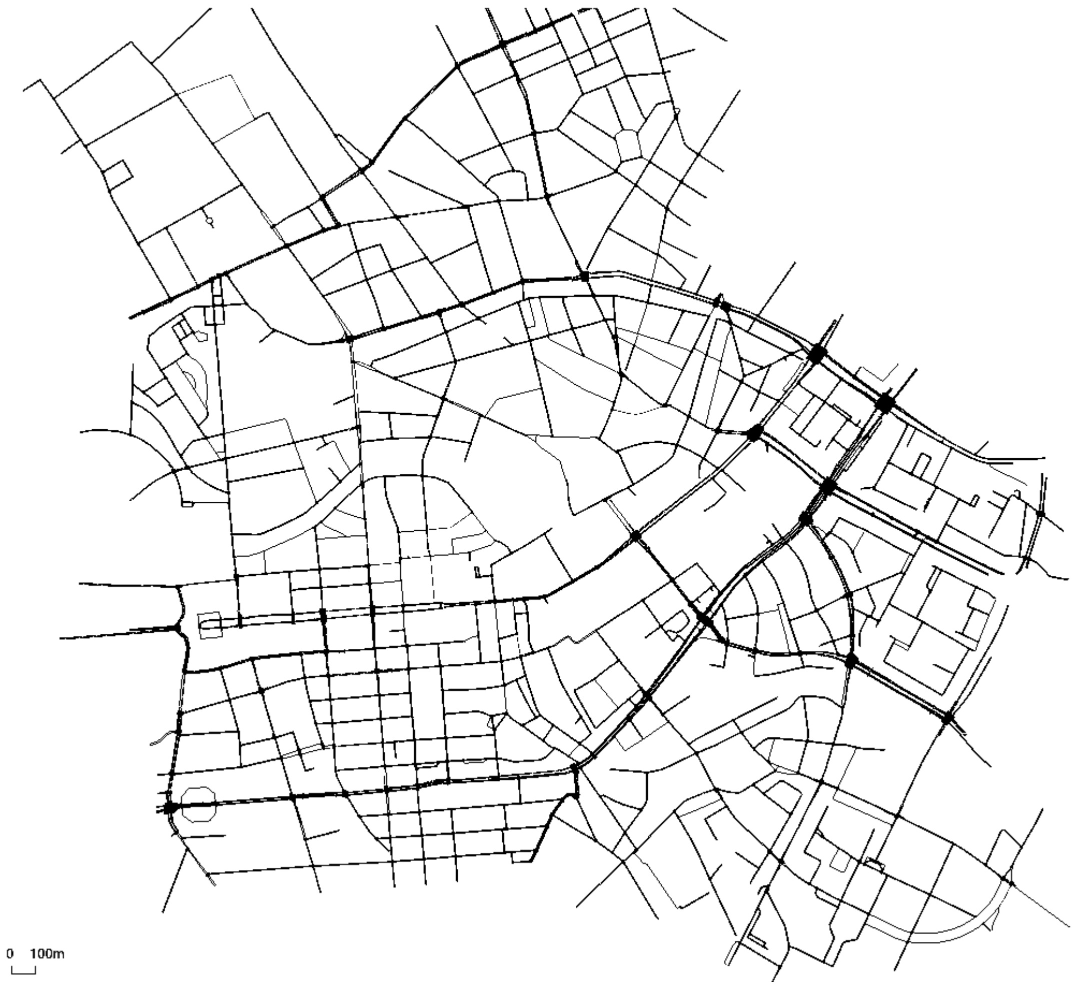

Figure 4: Screen shot of the Berlin center network used in this section.

Figure 5 shows the median delays for this set-up, once more using the four different methods as described in section 2:

fixed A run with a default settings of traffic signals with fixed cycle and split times.

actuated All the traffic lights are switched to actuated ones.

optSC Optimizing each individual intersection to the parameters put forward by Webster's approach, creating individual cycle times and splits. We chose to optimize every 6 hours, so at 0, 6, 12 and 18 'o clock the traffic lights switch its program to one adapted to the demand in this time-interval.

optSCO On top of the optimized split cycles, a simple co-ordination approach tried to change the offsets of the signals to improve the co-ordination between the signals. Note that we kept the individual cycle times here although this is not in favor of an advanced co-ordination. The co-ordination was performed based on the traffic flows, the respected time intervals and adapted cycle, and the split times corresponding to 'optSC'.

In general, all four controllers perform in a typical manner with peaks in delays at around 7-8 'o clock in the morning and 16-17 'o clock in the afternoon when demand is highest. The default method 'fixed' expectedly produces the highest delay during a whole day. Webster's approach 'optSC' improves the scenario significantly whereas the co-ordinated method 'optSCO' produces quite similar results as 'optSC'. The 'actuated' controller overall performs slightly better than the other methods.

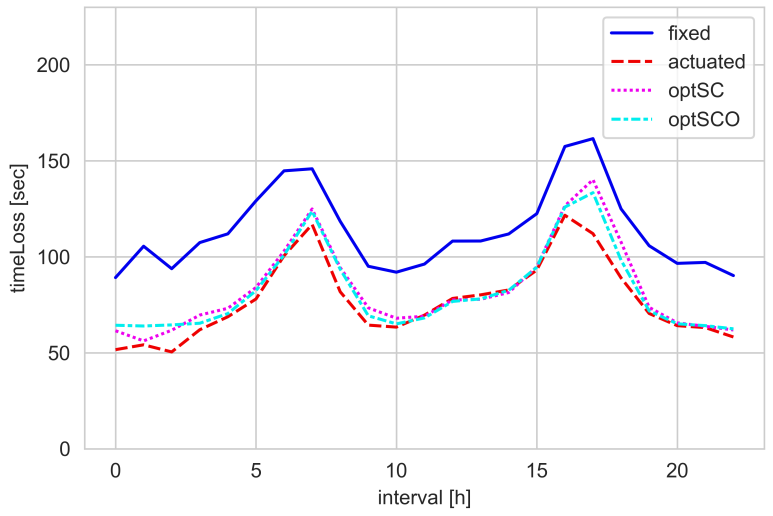

Figure 5: Median delays for one full day in a part of a real road network of Berlin simulated with four different methods of traffic light control.

The results of this real network investigation support the impressions from the theoretical research presented in this paper. The controller 'actuated' seems to be the most robust control because it adapts fasted to unexpected disruptions in traffic flow and demand. We observed that even solitary disruptions in the network produce large congestions along main roads and busy intersection which only dissolve after traffic demand declines. 'Actuated' outperforms the other methods because it adapts quicker to these conditions whereas 'optSC' and 'optSCO', although still significantly better than a 'fixed' traffic control, seem to perform best with low demands but not as robust for higher demands.

4 Caveats, conclusions, future work

Real road networks might be different. They are organized in a more hierarchical manner, i.e. there are few large roads that form a kind of backbone, and a lot of smaller roads surrounding them with considerably smaller load. Also, demand is not random, and they have at least some kind of user equilibrium. So, either a better model to create random networks is needed, or one may use parts of real networks instead of the randomly created ones to go beyond. Nevertheless, the networks investigated here are a much better substitute for real networks than the often very theoretical examples where just one arterial in one direction is co-ordinated [3].

Investigations done with a part of the real road network of Berlin (about 120 traffic signals), with a realistic demand point to the same direction, albeit the gain in actuated control is not always as big as the results shown here (see the appendix A).

References

[1] SUMO wiki. Last visited on 2019-01-29.

[2] Yun-Pang Flötteröd and Michael Behrisch. Improving sumo's signal control programs by introducing route information. In *SUMO 2018- Simulating Autonomous and Intermodal Transport Systems*, EPiC Series in Engineering, 2, pages 162–172, May 2018.

[3] N. H. Gartner and P. Wagner. Traffic flow characteristics on signalized arterials. *Transportation Research Records*, 1883:94–100, 2004.

[4] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[5] F. V. Webster. Traffic signal settings. Technical report, Road Research Laboratory, 1958.

[6] Bao-Lin Ye, Weimin Wu, and Weijie Mao. A method for signal coordination in large-scale urban road networks. *Mathematical Problems in Engineering*, 2015:15, 2015. Article ID 720523.

A APPENDIX – The two scripts

A.1 Coordination of traffic light systems

The algorithm implemented in `TLSCoordinator.py` uses a network file together with a traffic demand file (routes) to compute a network-wide traffic light coordination (a set of offsets). It works in three main steps.

1 Successive pairs of traffic lights are identified that are encountered by vehicles in the loaded traffic demand

2 The list of all pairs is ranked in descending order according to the following two numbers:

- Road priority of the edge at the start of the pair
- The total number of vehicles which encounter the pair divided by the travel time between the two traffic lights of the pair

3 The pairs are coordinated in a greedy manner according to their ranking using the following steps:

- If the two traffic lights are not yet part of the coordination, their offset is set to equal the travel time between them
- If the first traffic light is already part of a coordination, the offset of the second one is added to the end of the coordination
- If the second traffic light is already part of a coordination, the offset of the first one is added to the start of the coordination
- If both traffic lights are coordinated, their offsets remain unchanged

The resulting offsets are written to a file which can be loaded as part of the simulation input.

A.2 Computation of splits and cycle time

The `tlsCycleAdaptation.py` script uses a network file together with a traffic demand file (routes) to compute a network-wide set of green times (and subsequent cycle times) for all the traffic lights in a SUMO network, see [2] for more details.

This is done by a simple application of Webster's approach [5]: for each intersection, it computes the critical lane group flows q_i and the corresponding flow ratios (which are often named $\lambda_i = q_i/s_i$, where s_i is the saturation flow of the critical lane group i). Then, the optimum cycle time c can be computed:

$$c = \frac{1.5L + 5}{1 - \sum_i \lambda_i}$$

where L is the sum of the loss times.

In a second step, the green times g_i are computed by:

$$g_i = \frac{\lambda_i}{\sum_i \lambda_i}(c - L)$$

This is done for each intersection separately, resulting in a set of traffic light parameters for each of them. Note especially, that a separate cycle time may result for each intersection, they are only limited by a maximum cycle time which is a user-specified parameter.

It should be pointed out, that the resulting cycle times depend critically on the saturation flows. The script uses a default value of 0.5 veh/s (or 1800 veh/h for the saturation flow, which is not a bad approximation to SUMO's default simulation model. However, this might have different values in different settings (for instance for right and left turning). See the supplemental material in appendix C for a few more details on this. The script does not try to use different saturation flows for different traffic conditions, there is one saturation flow only.

The resulting green and cycle times are finally written to a file which can be loaded as part of the simulation input and then uses these traffic signal settings.

B APPENDIX – Supplemental Material

B.1 Webster and Actuated – Single intersection

This section analyses a single intersection, for which we have Webster's theory [5]. It is a four-arm intersection with two lanes per arm, and all the vehicles go straight; this eases the comparison with Webster's theory. Two hours have been simulated during which demand was hold constant; this has been repeated for different demands, to scan the Webster curve delay d as function of demand q. Note, that demand is not equal to the actual inflow, demand is what SUMO tries to insert, but inflow is what is finally being inserted in the simulated study area. This is made easier by SUMO's option `--max-depart-delay 90` which tries to insert generated vehicles for 90 seconds; after this time, they are thrown away. This, by the way, yields also a quite transparent method to determine the saturation flows for different conditions and vehicle parameters.

The single intersection has been run through four different scenarios, divided into ideal and realistic settings, and then into a fixed cycle control and an actuated control. Ideal consists

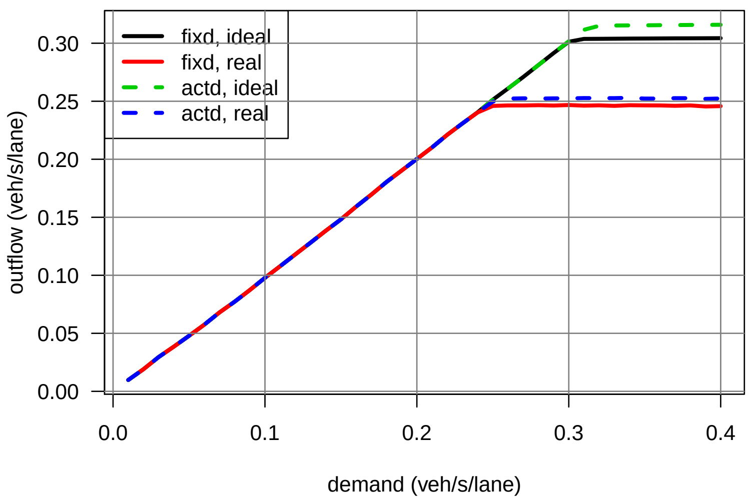

Figure 6: Outflow versus demand for the four different set-ups. The saturation flow itself can be computed for the fixed cycle controler by multiplying the maximum outflow measured with the inverse green split, which is this case was $c/g = 76/35 \approx 2.17$.

of identical vehicles with `speedDev = 0` and `sigma = 0`, the real(istic) version is `speedDev = 0.1` and `sigma = 0.5`. (In principle, one should correct the acceleration a bit for the increase of `sigma`, this is not done here.)

By holding all the traffic signal parameters constant and only changing demand, we arrive at a outflow versus demand curve, where saturation flow can be computed from the plateau to be reached for large demand, see Figure 6 for an example. This is named outflow here, since it is measured at the downstream end; however, the set-up makes sure, that the outflow measured is equal to the inflow into the study area.

Note, that the Figure 6 also shows, that the actuated control is outperforming a fixed cycle controler even at saturation, albeit by a small amount of 2.5 % only.

Next, the delay versus demand curve is analyzed, Figure 7 displays the result. As function of demand, delay increases until it finally reaches capacity and delay (and along with it, outflow) no longer increases. This is different from Webster's theory, where delay goes to infinity, but the simulation result is much more realistic: once a vehicle has entered the simulation, it is guaranteed to finally leave, after a finite time which can be computed to be roughly the number of vehicles on each link multiplied by the time each vehicle needs to leave minus the ideal travel time L/v (where L is the link length, v is the speed), i.e. $d_{\max} = \frac{L}{\ell}\frac{c}{g\,s} - \frac{L}{v}$, with ℓ the vehicle length, g the green time, c the cycle time, and s the saturation flow.

The delay d versus demand q curve should follow Webster's theory:

$$d(q) = \frac{c(1-\lambda)^2}{2(1-\lambda x)} + \frac{x^2}{2q(1-x)} - 0.65\left(\frac{c}{q^2}\right)^{1/3} x^{2+5\lambda} \qquad (1)$$

where $x = q/(\lambda s)$ is the degree of saturation, s is the saturation flow of the approach, c is the cycle time, and $\lambda = g/c$ is the green split, with g the green time. In principle, c and g are known from the simulation, and s can be measured by the approach above.

These numbers have a certain uncertainty (the green time is a so called effective green time, the saturation flow is an idealized quantity which is best for longe cycle times, the theory

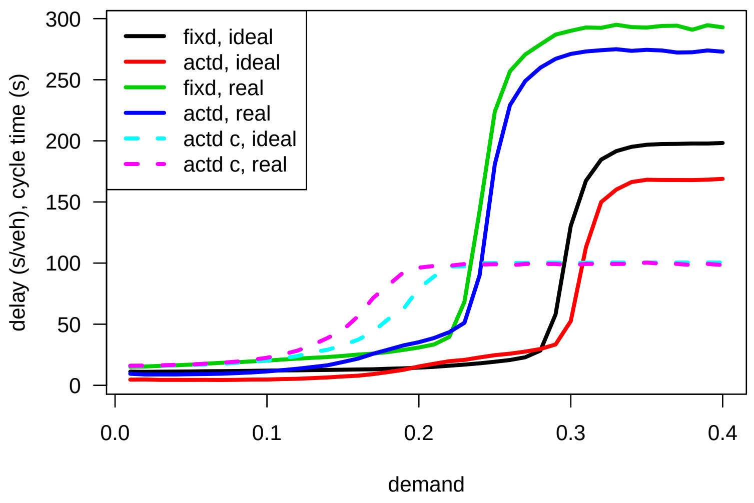

Figure 7: Delay versus demand for the four scenarios described in the text. The broken lines are the cycle times of the actuated controler for the ideal and the realistic case.

behind is roughly a M/D/1 queue, which is not what the simulation is), so another approach (often used in traffic) is to fit the theoretical curve to the simulated one. To do so, a cut-off has to be introduced in Webster's equation to avoid infinities which does not work well with non-linear fitting algorithms. Still, the result is (so far) not overwhelming, the Figure 8 shows one result. Note the golden line which has been created by using slight manual modifications to the non-linear fitting results.

B.2 Arterial test

This is to test that the scripts `tlsCycleAdaptation` and `tlsCoordinator` used above work correctly.

To simulate an ideal arterial, perfect vehicles are used. They have zero sigma, and zero `speedDev`. The arterial consists of an input and an output link of 200 m length, and a variable number of links between them with the length 375 m, creating a travel time of 27s from one signal to the next. Input flow is binomial, and from this, the optimal solution is (almost) clear: the optimal offset ϕ between signals is just 27 s, at least for the one direction in which there is input flow. The flow is just generated along this single one arterial, and in one direction only. Then, the following procedure is performed:

- Create the networks, one base network without traffic signals, and then a set of additional networks, four with fixed cycle control and one with a set of actuated signals. Finally, create the demand, by using a flow file and the base network as input for the `duarouter`. Note, that because of the stochastic input on the first link, the platoons produced by the first traffic light have unequal length.

- The difficult part is the cycle time; this is currently solved by using `tlsCycleAdaptation` to compute the maximum cycle time. However, this may not be the optimum cycle time for this set-up.

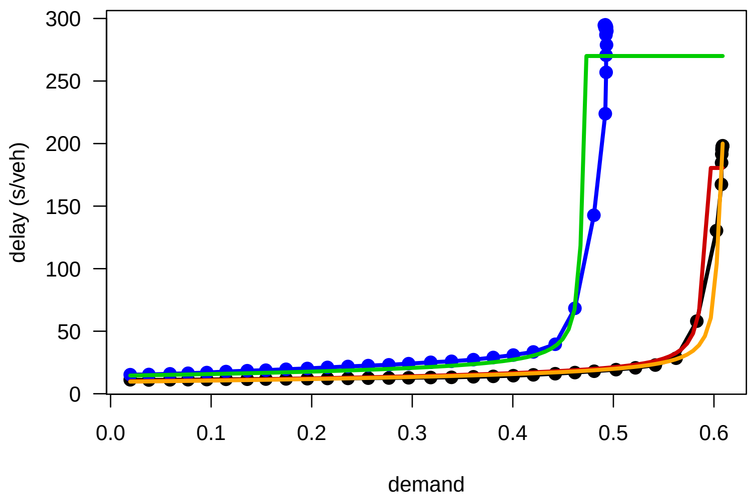

Figure 8: Comparison of the simulated delay curves with Webster's curves; black and blue are the simulation results, red and green are the results of non-linear fits to these simulation data, and the golden curve is manually tweaked to yield a visually more appealing fit. Note, that only the first two terms of Webster's equations have been used here.

- Then, the following five configurations are run, and the resulting delay times (mean and quantiles) are recorded:

  fix SUMO's default, but with adopted cycle times extracted from `tlsCycleAdaptation`. Note: no green-times are used, only the maximum cycle time.

  opt From this, the optimum solution (including green times) from `tlsCycleAdaptation`, together with the optimum set of offsets as computed manually.

  SC The optimum set of signal settings as computed by `tlsCycleAdaptation`.

  SCO The optimum set of signal settings as computed by `tlsCycleAdaptation`, together with the optimum offsets as computed by as computed by `tlsCoordinator`

  actd The result of the actuated system.

The results are detailed in Table 1 (demand was 1445 veh/h, 9 signals):

Name	$c\,(s)$	$\langle d \rangle\,(s/veh)$	$\min\{d\}\,(s/veh)$
fix	30	280.4	7.6
opt	55	24.6	0
SC	55	56.9	3.6
SCO	55	26.6	0
actd	22.5	37.7	3.3

Table 1: Results of the five methods regarding the delay time for ideal vehicles.

This demonstrates that under the set-up used here, the optimal solution is only slightly better than the combination 'SCO' of `tlsCycleAdaptation` and `tlsCoordinator`. Both approaches produce vehicles that do not have to stop at all when running along these 9 signals.

For this combination, the actuated approach ranks number 3, while optimizing the splits and cycle times only is forth.

Note, that with shorter arterials, the gain becomes smaller, it might even happen, that the gain of the first signal by the actuated controler outperforms the gains of the subsequent green wave and the actuated solution is the best, which is especially true if the cycle time is not optimized – the green wave seems to become better for longer cycle times. Of course, when excluding this first intersection[1] by looking at a pure green wave corridor, then `tlsCoordinator` does a good job.

Note, that there might be better combinations of c and ϕ than the ones used here that depend on the number of signals, the demand, and so on.

B.3 Less ideal vehicles

Setting `sigma=0.5` and `speedDev=0.1` makes the delay times worse, and all the zero delay times vanish, i.e. no vehicle runs through the sequence of signals without stopping once. However, the ranking between the methods remains robust.

Name	$c\,(s)$	$\langle d \rangle\,(s/veh)$	$\min\{d\}\,(s/veh)$
fix	30	411.4	13.3
opt	55	46.6	10.2
SC	55	74.8	17.1
SCO	55	51.0	10.1
actd	22.5	60.7	17.7

Table 2: Results of the five methods regarding the delay time for more realistic conditions.

[1] But why should this be done? In real world, a co-ordination has to start somewhere anyway.