Synchronization of traffic signals through a heuristic-modified genetic algorithm with GLTM

Guido Gentile, Daniele Tiddi
Dipartimento di Idraulica Trasporti e Strade, Sapienza Università di Roma

Abstract

Urban signal timing is a non-convex NLP problem. Finding an optimal solution on not very small and simple networks may take long time, wherever possible. The present paper focuses on signal synchronization, thus creating fast-flow corridors on one or more network road arterials. To do this, a genetic-like algorithm is applied, in which new solutions generation follows heuristic conceptions. This can be carried out thanks to the specific formulation adopted, suitable for synchronization problems. The objective function is evaluated by the General Link Transmission Model, a very fast macroscopic dynamic simulator referring to the kinematic waves theory. Through this, queues dynamic evolution, spillback phenomenon and vehicles travel times are explicitly taken into account.

Key-words: traffic control, signal setting, synchronization, optimization, genetic algorithm, general link transmission model.

1. Introduction

Urban traffic is mainly characterized by ground-level intersections, where manoeuvres of drivers (and pedestrians, too) often conflict. Traffic signal allows them, alternating conflicting manoeuvres. Traffic signals settings are determined by real-time or fixed time schemes. The first procedure needs real-time traffic surveys and an algorithm subsequently calculating the best solution basing on detected flows. The second one determines the best signal settings on a given time period, basing on demand flows obtained by historical surveys. A mixed procedure has been sometimes adopted, selecting real-time the best signal settings inside a given set previously built. Finding a fast, effective, real-time algorithm is nowadays still an hard task, so heuristic methods often occur. The algorithm here proposed finds a sub-optimal solution for a given demand data, synchronizing traffic signals. Synchronization is a specific traffic signal setting problem: it consists in coordinating the timing of successive intersections along one specific route (or more, if not competing), to improve the movement of vehicles through this path.
A traffic signal is described by its timing variables. First of all, the time duration of its lights (red, yellow/amber, green). The sum of these times expresses the signal *cycle*, i.e. the time period before the same light turns on again. Having the cycle time, lights duration is often expressed as its *split*, i.e. the ratio of cycle time. Considering more signals, we need to introduce the *offset*, i.e. the time period between a common reference instant and the cycle start. To avoid hypothetically any stop along a path, traffic signals should have identical cycle and green times and offset equal to the travel time from one signal to the next one. The time period in which every light of one signal is kept unchanged is called *phase*. A phase thus enables a specific set of non-conflicting manoeuvres amongst all possible manoeuvres on the intersection.
Depending on goal the problem formulation and thus the solving algorithm vary subsequently. Traffic lights improve drivers safety, but they unavoidably introduce delays in travel times, too. So, minimizing total delay, as sum of all vehicles delays, may be an intuitive objective. Total delay minimization is a non-convex problem, so global optimum can not be found analytically. On the

other side, another goal in the past was maximizing the bandwidth, i.e. the number of vehicles able to run their path without stops. Bandwidth maximization is a quasi-concave problem, thus analytic solving algorithms can be used to find the optimum; unfortunately, it does not take into account demand data, so it is a good-engineering practice, but it yields not minimum delay. Further goals can be minimizing traffic congestion, vehicles' number of stops, vehicle externalities or multi-criteria functions.

First approaches to traffic synchronization were minimizing separately each intersection's cycle time, constrained to satisfy respective flows, then adopting the maximum cycle time found, or maximizing network capacity, i.e. ordinary demand matrix multiplier, keeping the intersections under-saturated. Cohen (1983) used maximum bandwidth solutions as starting points of its optimization. Cohen and Liu (1986) constrained the minimum delay problem with maximum bandwidth solutions. Messer *et al.* (1987) used a Passer-II simplified node delay model to maximize bandwidth. The work has been later extended by Malakapalli and Messer (1993), expressing nodes delay as a function of the difference between offsets of contiguous nodes. Gartner et al. (1991) and later Gartner and Hou (1994) developed a flows-dependant objective function, to maximize bandwidth. Hadi and Wallace (1993) first introduced a genetic algorithm using Transyt to maximize bandwidth. Park et al. (1998) used a genetic algorithm and mesoscopic simulator to optimize traffic signal settings for oversaturated intersections. De Schutter (2001) formulated the problem as a complementarity problem. Dazhi et al. (2006) proposed a bi-level programming formulation and a heuristic solution approach for dynamic demand with user stochastic route choice. Ying *et al.* (2007) minimized total travel time using a sensitivity analysis algorithm.

2. Formulation

We aim to minimize total delay. To do this, it is useful remarking that users total travel time can be expressed as sum of two terms: the free flow travel time on the non-signalized network, which is a constant term, and an additional delay due to the interaction between flows and traffic signals, i.e. congestion, stops and queues. Considering this, it is immediate that minimizing total delay and total travel time are equivalent problems. On urban networks most delays are spent on main arterial roads, as there are larger flows and thus higher congestion. So, our objective is fastening flows along these corridors.

Total delay is a non-convex function, and optimal signal setting feasible set comprehends non-linear constraints and sometimes integer variables (phases optimization). More, as dynamic assignment is simulated and has not an analytical formulation, no derivative is available, nor any convexity information. So, finding a global optimum can not be carried out by analytic optimization algorithms. In such problems, genetic algorithms are considered efficient methods to determine sub-optimal solutions.

We assumed the following simplifications. First, only two lights signals are considered: an *effective green*, indicating the time period vehicles can actually cross the intersection, and a residual red, when vehicles intersection crossing is prohibited. Second, phases definition and ordering is supposed as fixed. Obviously, synchronization requires cycle time to be the same for all signals. The first simplification is very common in traffic works; the second one is often adopted to neglect the discrete optimization problem of phase setting; the third one is intrinsic in synchronization itself. The main assumption comes from focusing on synchronized path. We define *main phase* all signal phases allowing the flow to run along the defined path and *secondary phase* any other one. We focus on main phases, calculating subsequently the secondary phases green splits through an heuristic rule. So, the problem's variables are the common cycle time and green and offset of main phases of every synchronized intersection. Having more than one corridor is possible, considering every corridor main phases. Obviously no "concurrent" couple of corridors may exist. Without any loss of generality we will refer to one single corridor.

We introduce the following notation:

$TD(f)$	total delay time
TTT(f)	total travel time
FFTT	free flow travel time (constant)
s	number of synchronized intersections
c	cycle (unique)
g_i	i-th intersection main phase green time
o_i	i-th intersection main phase offset
nP_i	i-th intersection number of phases
$g_i^{\,p}$	i-th intersection green time of phase p ($g_i^{\,1} = g_i$)
$o_i^{\,p}$	i-th intersection green time of phase p ($o_i^{\,1} = o_i$)
f	traffic flows
c_{min}	minimum cycle
c_{max}	maximum cycle
g_{min}	minimum green time
g_{max}	maximum green time

The problem can be written as:

$$\min_{c,\vec{g},\vec{o}} TD(f) \equiv \min_{c,\vec{g},\vec{o}} TTT(f) = TD(f) + FFTT$$

$$s.t. TTT(f) \text{ from dynamic assignment with traffic signal settings } (c,\vec{g},\vec{o})$$

$$c \in [c_{\min}, c_{\max}]$$

$$g_i \in [g_{\min}, g_{\max}] \quad \forall i \in 1..s$$

$$o_i \in [0, c) \quad \forall i \in 1..s$$

This way, it is possible to reduce problem complexity, having no more $2{\cdot}s{\cdot}m$, having m the average number of phases per intersection, but just $2{\cdot}s$ variables (green and offset of every intersection main phase). Every solution is in fact composed by $2{\cdot}s$ + 1 variables ($c, g_1, o_1, \dots, g_s, o_s$).

2.1 Secondary phases heuristic

Having the main phases signal settings, calculating secondary phases ones is outperformed by a good-engineering practice. Residual green can be split among secondary phases either by a predefined proportion or by the Webster's equisaturation rule, i.e. proportionally to each phase of that intersection historical flows. Secondary phases' offset comes subsequently, as the phase sequence is supposed to be fixed, through the following algorithm:

```
for i = 1 to s do
        for p = 2 to nPi do
                oi(p) = oi(p-1) + gi(p-1)
                if oi(p) >= c then oi(p) = oi(p) - c
```

3. The dynamic traffic model

Every solution's quality is evaluated by a macroscopic dynamic traffic model, the General Link Transmission Model (Gentile, unpublished). This model improves the Cell Transmission Model (Daganzo, 1994) idea, adopting the Simplified Theory of Kinematic Waves, but not discretizing the space into cells, considering the link as a channel with bottlenecks on enter and exit. The entering bottleneck capacity is reduced by the flow on the link, if equal to the link capacity, while the exiting bottleneck is traffic signal controlled; link performances depend by flow on it. The whole model substantially bases on an algorithm returning link flows travel time, for a given entering time and cumulated entering and exiting flows.

3.1 Link performances

Let $q(x, \tau)$ be the flow of vehicles crossing point $x \in [0, L]$ at time τ, and let $t(\tau)$ be the leaving time of a vehicle arriving to the element at time τ. The cumulative flow $Q(x, \tau)$ is given by:

$Q(x, \tau) = {}_0\int^{\tau} q(x, \sigma) \cdot d\sigma$.

Thus, based on the fluid paradigm, the FIFO rule holds, and can be expressed formally as:

$Q(0, \tau) = Q(L, t(\tau))$.

On this basis, once the cumulative flow temporal profiles at the initial and end points of any element, or series of elements, are known, the exit time temporal profile can be easily determined.

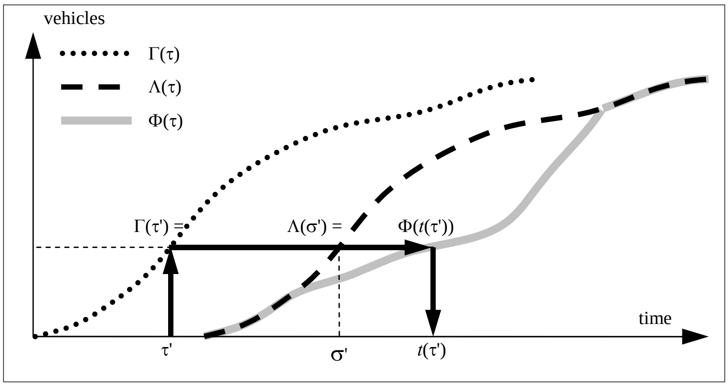

Figure 1: Computation of the link exit time based on the cumulative leaving flow from the initial bottleneck and the cumulative exit flow from the link by applying the FIFO rule.

The solution of kinematic waves equations is based on time discretization in adjacent intervals $(\tau_{i-1}, \tau_i]$, with $i = 1, \ldots, n$. Under the classical numerical approximation that the flows are constant during each interval, we can apply the following algorithm, where we assume that

$Q(0, \tau_n) = Q(L, \tau_n)$,

$Q(0, \tau_0) = Q(L, \tau_0) = 0$,

and T_0 is the free flow travel time of the element, or series of elements.

```
t(τ0) = τ0 +T0
j = 1
for i = 1 to n do
   until Q(L, τj) ≥ Q(0, τi) do j = j + 1          (A)
   if Q(0, τi) = Q(0, τi-1) then
      t(τi) = τi + T0
      if t(τi) < t(τi-1) then t(τi) = t(τi-1)
   else
      t(τi) = τj-1 + [Q(0, τi) - Q(L, τj-1)] · (τj - τj-1) / [Q(L, τj) - Q(L, τj-1)]          (B)
```

The input of the algorithm is $Q(0, \tau_i)$ and $Q(L, \tau_i)$, while the output is $t(\tau_i)$, for $i = 0, \ldots, n$. The cycle *(A)* aims at finding, for each instant τ_i in chronological order, the earliest instant τ_j such that:

$Q(L, \tau_{j-1}) < Q(0, \tau_i) \leq Q(L, \tau_j)$

Since the leaving flow is by definition constant during the interval $(\tau_{j-1}, \tau_j]$, the cumulative leaving flow increases linearly with slope:

$[Q(L, \tau_j) - Q(L, \tau_{j-1})] / (\tau_j - \tau_{j-1})$

Therefore, in the general case where $Q(0, \tau_i) > Q(0, \tau_{i-1})$, the exit time $t(\tau_i)$ results from the simple proportion in *(B)*. In the particular case where no flow arrives at the element in the interval $(\tau_{i-1}, \tau_i]$, the exit time $t(\tau_i)$ may be undetermined; it is thus set by definition as the maximum between the free flow exit time $\tau_i + T_0$ and the exit time $t(\tau_{i-1})$.

3.2 Bottlenecks

Bottlenecks allow simulating explicitly the formation and dispersion of vehicle queues, thus letting to evaluate the delay due to the presence of intersections, which is an important part of the total travel time in highly congested urban networks. To represent the spillback phenomenon, we assume that each link is characterized by two time-varying bottlenecks, one located at the initial point and the other one located at the end point, called "entry capacity" and "exit capacity", respectively.

The entry capacity, bounded from above by the physical capacity which is typically related to the number of road lanes, is meant to reproduce the effect of queues propagating backwards from the end point of the link itself, that can reach the initial point so inducing spillback conditions on the upstream links. In this case the entry capacity is set to limit the current inflow at a value that keeps the number of vehicles on the link equal to the storage capacity currently available, which is related to the queue density along the link. The latter changes dynamically in time and space as a function of the outflows at previous instants. Specifically, any change in the rate of the space freed by vehicles exiting the link at the head of the queue takes some time to become actually available at the tail of the queue, while the jam density multiplied by the length is just the upper bound of the storage capacity, which can be reached only if the queue is not moving.

The exit capacity, bounded from above by the saturation capacity which is typically related to the regulation of the road intersection, is meant to reproduce the effect of queue spillovers propagating backwards from the downstream links, which in turn may generate hypercritical flow states on the link itself. For given inflows, outflows and intersection priorities, the exit capacities are obtained as a function of the entry capacities based on flow conservation at the node.

Based on the Newell-Luke Minimum Principle the cumulative flow leaving the bottleneck at time τ is the minimum among each cumulative outflow that would occur if the queue began at a previous instant $\sigma \leq \tau$, that is:

$$Q(L, \tau) = \min\{Q(0, \sigma) + \Theta(\tau) - \Theta(\sigma): \sigma \leq \tau\} ,$$

where $\Theta(\tau)$ is the cumulative bottleneck capacity at time τ, that is:

$$\Theta(\tau) = \int_0^\tau \theta(\sigma) \cdot d\sigma .$$

The above expression states that if there is no queue at a given time τ, the cumulative leaving flow $Q(L, \tau)$ is equal to the cumulative arriving flow $Q(0, \tau)$. If a queue arises at time $\sigma < \tau$, from that instant until the queue will eventually vanish, the outflow equals the bottleneck capacity, and then the cumulative leaving flow $Q(L, \tau)$ at time τ results from adding to the cumulative arriving flow $Q(0, \sigma)$ at time σ the integral of the bottleneck capacity between σ and τ, that is $\Theta(\tau) - \Theta(\sigma)$. Notice that, if there is no queue at time τ, the cumulative leaving flow is the same of the case when the queue arises exactly at $\sigma = \tau$.

Figure 2 depicts a graphical interpretation of the general equation for a bottleneck with a time-varying capacity, where the temporal profile $Q(L, \tau)$ of the cumulative leaving flow is the lower envelop of the following curves: a) the cumulative arriving flow $Q(0, \tau)$; b) the family of functions $Q(0, \sigma) + \Theta(\tau) - \Theta(\sigma)$ with $\tau > \sigma$, for every time σ, each one obtained as the vertical translation of the temporal profile relative to the cumulative bottleneck capacity that goes through the point $(\sigma, Q(0, \sigma))$. No queue is present when curve a) prevails; therefore, the queue arises at time σ' and vanishes at time σ''.

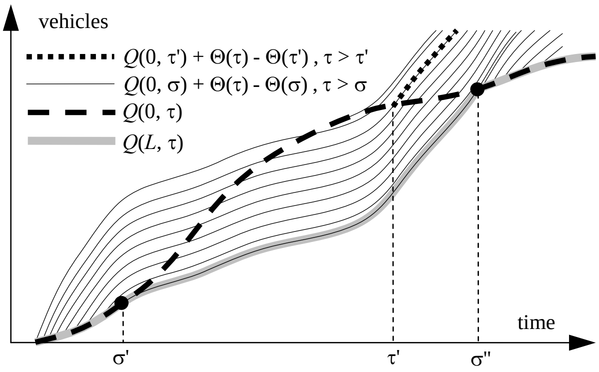

Figure 2: Bottleneck with time-varying capacity.

Let $N(\tau)$ be the number of vehicles queuing to exit the bottleneck at time τ, that is:
$N(\tau) = Q(0, \tau) - Q(L, \tau)$.
The equation can be numerically solved by means of the following algorithm:

```
N(τ_0) = 0
for i = 1 to n do
   N(τ_i) = N(τ_{i-1}) + Q(0, τ_i) - Q(0, τ_{i-1})
   if N(τ_i) ≤ Θ(τ_i) - Θ(τ_{i-1}) then
      Q(L, τ_i) = Q(L, τ_{i-1}) + N(τ_i)
      N(τ_i) = 0
   else
      Q(L, τ_i) = Q(L, τ_{i-1}) + Θ(τ_i) - Θ(τ_{i-1})
      N(τ_i) = N(τ_{i-1}) - Θ(τ_i) + Θ(τ_{i-1})
```

The input of the algorithm is $Q(0, \tau_i)$ and $\Theta(\tau_i)$, while the output is $Q(L, \tau_i)$ and $N(\tau_i)$, for $i = 0, \dots, n$. The number of vehicles desiring to leave the bottleneck during the interval $(\tau_{i-1}, \tau_i]$, for short called here the demand, is given by the number of vehicles $N(\tau_{i-1})$ queuing to exit the bottleneck at the beginning of the interval, plus the number of vehicles $Q(0, \tau_i) - Q(0, \tau_{i-1})$ that arrive at the bottleneck during the interval. But, due to the capacity constraint, only the number of vehicles $\Theta(\tau_i) - \Theta(\tau_{i-1})$, for short called here the supply, can at the most exit the bottleneck. If the supply is higher than the demand, then all such vehicles will actually leave the bottleneck during the interval, and no vehicle will be queuing to exit the bottleneck at the end of the interval; otherwise, only a number of vehicles equal to the supply will leave the bottleneck during the interval, and the rest of the demand will be queuing to exit the bottleneck at the end of the interval.

4. Genetic algorithm

Genetic algorithms are efficient finders of good solutions: they are heuristic methods determining sub-optimal solutions through a heuristic exploration of the space of solutions. They require to evaluate every selected solution point, so they are particularly useful when its evaluation requires low computational times and no derivative information for a faster convergence can be used. Genetic algorithms name is due to the parallelism between their scheme and biologic phenomena studied by Genetics; actually, they take many terms from this discipline.

4.1 Initial population

Genetic algorithms start from a feasible solutions subset, called *initial population*, composed by r feasible solutions. One solution, called *individual*, is described by its *genes*, i.e. the problem variables. In our study every individual is a specific signal setting, whose $2 \cdot s + 1$ genes are the common cycle time and each intersection green and offset. Each algorithm iteration is called *generation*, at the end of which only most suitable individuals are selected to breed the next

generation: suitability is evaluated by a *fitness function*. Fitness function is the optimization problem objective function. Without any loss of generality, from now on we will refer to the fitness function as the total delay on the network, we aim to minimize. Through the Darwinian “survival of the fittest” rule, each generation a set containing individuals better than previous ones is obtained, as the best individual objective function is implicitly equal or less to the one of previous generation’s best individual. This way only solutions more capable to minimize traffic delays are allowed to “survive”.
Initial population strongly affects the algorithm convergence speed. Often, it is created simply selecting *r* random feasible solutions, forsaking the purpose of increasing its speed. We instead chose to breed it with maximum bandwidth solutions. The maximum bandwidth solutions are created through the MAXBAND equivalent systems algorithm (Papola and Fusco, 1998); this returns optimal offsets, for given cycle time and intersection green times. To obtain *r* solutions, different input variables are evaluated: cycle time interval $[c_{min} , c_{max}]$ is split into regular intervals, for each cycle value, three green settings are taken:

- pre-defined green splits (ex.: actual signal settings);
- maximum green to main phase (accordingly with secondary phases minimum constraints);
- green splits according to Webster’s equisaturation rule.

These are not optimal solution, but allow to start from quite good solutions.

4.2 Space of solutions exploration

Starting from initial population, genetic algorithms explore the feasible set generating new solutions through heuristic procedures and evaluating them. New solution points are produced by *mutation* and *crossover* operations. Mutation consists in selecting one individual and modifying one or more genes. How the modify is carried out on the single gene depends on its coding: if a binary coding is adopted then mutation switches one or more bits, if it is a continuous value then mutation increases or decreases it. In this case, genes c, g_i and o_i, are expressed as real numbers: each generic variable x is randomly increased or decreased of a quantity between 0 and x_{int}, where x_{int} is the maximum accepted variation.

```
for j = 1 to nCh do
    Δx = (2 * rnd - 1) * x_int
    x = x + Δx
    if x < x_min then x = x_min          (C)
    if x > x_max then x = x_max          (D)
```

The mutated solution is adjusted by the boundary constraints checks *(C)* and *(D)* to produce a solution belonging to the feasible set. Usually every existing solution is mutated with a very low probability p_{mut} , between 0.001 and 0.05: in this case the original solution is lost. Differently, we produce *t* new individuals starting from selected ones’ genes, creating a mutated copy of original individual. This allows to not loose previous solutions and contemporaneously to control the mutation rate through the *t* parameter.
The crossover consists in selecting two target individuals, denoted as *parents*, to generate one, sometimes two, *son* individual; the son’s gene is inherited either from one or other parent, randomly. The following is the algorithm doing this:

```
for j = 1 to s
    if rnd < 0.5 then
        x_j = P1.x_j
    else
        x_j = P2.x_j
    If x_j * c < x_min then x_j = x_min / c
    If x_j * c > x_max then x_j = x_max / c
```

where P1 and P2 are the individual's parents. Individuals probability to be selected as parents is proportional to their fitness function: we can expect the combination of better solutions to be statistically more fitting than worse ones. This is usually not granted: it depends on the crossover operation actual meaning. Inheriting *i*-th gene means assuming the *i*-th intersection traffic signal setting (except for the 0-th gene, the cycle time). Thus, taking one intersection traffic signal setting from a much better solution is supposed to return a better behavior of that intersection. Contemporaneously, this behavior is controlled by two parameters, G_i and O_i, and inheriting just one of them would not keep anything about the parent fitness: for this reason the couple (G_i , O_i) is entirely inherited in crossover.

More, each crossover step two mirror-sons are generated: each parental gene goes to one or the other son. Parents may be automatically discarded or may be not. Similarly to the mutation process, we chose to save parents and to establish to create every generation *u* new sons individuals through crossover operations. This allows us to introduce the parameter *u* as crossover evolution rate.

When the new solutions creation stops, a subset of available solutions is selected to produce next generation set, while remaining are lost. The selection is usually done randomly, assigning more surviving probabilities to best individuals. Sometimes best *v* solutions directly get surviving probability equal to 1: this technique is called *elitism* and grants best found solutions to be never lost. In this study the technique is taken to the extreme, as $v = r$, the population dimension. I.e. exactly the best *r* individuals are selected to compose next generation, while remaining $t + u$ solutions are deleted.

5. Conclusions

Reduced computational cost, in memory and time terms, is this algorithm main evidence. This allows to quickly obtain the best cycle time, green times and offsets of a set of synchronized traffic signals along a traffic corridor. The algorithm can be used offline to determine the best solution for one single scenario, i.e. for a given demand data. Unfortunately not only traffic flows, but demand has a dynamic evolution, too. So what is best now can be far from the best solution some hours (or some minutes!) later. Plan-selection control strategies select real-time the best synchronization settings amongst one set of possible strategies, previously established: the evaluation is based on traffic real-time surveys to build demand data. The proposed algorithm can be used to populate the available solutions set, as well as the General Link Transmission model to evaluate available plans.

But this would still not take full advantage of its quickness. More advanced control strategies completely determine the new synchronization settings, basing on surveys. The setting is then updated every time interval, from some dozen of seconds to some minutes. This can be done through the genetic algorithm using real-time demand data and finding quickly best settings. The same could not be done by previous approaches due to too expensive computational times.

References

1. Ceylan H., Bell M.G.H., *Genetic algorithm solution for the stochastic equilibrium transportation networks under congestion*, Transportation Research Part B, 2005, vol. 39, pp. 169-185
2. Ceylan H., Bell M.G.H., *Traffic signal timing optimization based on genetic algorithm approach, including drivers' routing*, Transportation Research Part B, 2004, vol. 38, pp. 329-342
3. Chang T., Sung G. (2004), *Modeling and Optimization of an Oversaturated Signalized Network*, Transportation Research Part B, 38, pp.687-707
4. Cohen S.L. (1983), *Concurrent Use of MAXBAND and TRANSYT Signal Timing Programs for Arterial Signal Optimization*, Transportation Research Rec., 906, 81-84
5. Cohen S.L. and Liu C.C. (1986), *The Bandwidth-Constrained TRANSYT Signal-Optimization Program*, Transportation Research Rec., 1057, 1-7

6. Colombaroni C., Fusco G., Gemma A. (2009), *Optimization of Traffic Signals on Urban Arteries through a Platoon-Based Simulation Model*, Proceedings of the 11th WSEAS International Conference on Automatic control, Modelling and Simulation, Istanbul, Turkey, May 30 - June 1, 2009
7. Daganzo C.F. (1994), *The cell transmission model: a dynamic representation of highway traffic consistent with hydrodynamic theory*, Transportation Research Part B 28, 269-287
8. Daganzo C.F. (1995), *The cell transmission model, part II: network traffic*, Transportation Research Part B 29, 79-93
9. Daganzo C.F. (1997) *Fundamentals of transportation and traffic operations*. Pergamon, Oxford, UK, chapter 4
10. Dazhi S., Benekohal R.F., Waller S.T. (2006), *Bilevel Programming Formulation and Heuristic Solution Approach for Dynamic Traffic Signal Optimization*, Computer-Aided Civil and Infrastructure Engineering, 21, pp. 321-333
11. De Schutter B., *Optimizing acyclic traffic signal switching sequences through an Extended Linear Complementarity Problem Formulation*, European Journal of Operational Research, 2002, vol. 139, pp. 400-415
12. Dion F., Rakha H., Kang Y., *Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections*, Transportation Research Part B, 2004, vol. 38, pp. 99-122
13. Feldman O., Maher M., *Optimization of traffic signals using a cell transmission model*, School of the Built Environment&Transport Research Institute
14. Fusco G, Gentile G., Meschini L., *et al.*, *Strategies for signal setting and dynamic traffic modelling*
15. Gartner N.H., Hou D.L. (1994), *Performance Evaluation of Multi-Band Progression Method*, 7th IFAC/IFORS Symposium on "Transportation Systems: Theory and Application of Advanced Technology", Tianjin, China, 24-26 August, 981-986
16. Gentile G., Meschini L., Papola N. (2005), *Macroscopic arc performance models with capacity constraints for within-day dynamic traffic assignment*, Transportation Research Part B 39, 319-338
17. Gentile G., Meschini L., Papola N. (2007), Spillback congestion in dynamic traffic assignment: a macroscopic flow model with time-varying bottlenecks, to appear on Transportation Research Part B
18. Malakapalli M.P. e Messer C.J. (1993), *Enhancements to the PASSER II-90 Delay Estimation Procedures,* Transportation Research Rec., 1421, 94-103
19. Mirchandani C., *Adaptive Traffic Management Systems using Programmable Devices*, Systems Engineering, 2002, Test & Evaluation Conference
20. Newell G.F. (1993), *A simplified theory of kinematic waves in highway traffic, part I: general theory; part II: queuing at freeway bottlenecks; part III: multi-destination flows*, Transportation Research Part B 27, 281-313
21. Papola N, Fusco G., *Maximal bandwidth problems: a new algorithm based on the properties of periodicity of the system*, Transportation Research Part B, 1998, vol. 32, no. 4, pp. 277-288
22. Papola N., G.Fusco (2000), *A new analytical model for traffic signal synchronization*, Proceedings of the 2nd ICTTS Conference, Beijing, China, 31 July-2 August, 2000
23. Park B., Messer C.J., Urbanik II T. (1999), *Traffic Signal Optimization for Oversaturated Conditions: Genetic Algorithm Approach*, Transportation Research Rec., 1683, pp.133-142
24. Wong S.C., Yang H., *Reserve capacity of a signal-controlled road network*, Transportation Research Part B, 1997, vol. 31, no. 5, pp. 397-402
25. Ying Q.Y., Lu H., Shi J., *An algorithm for local continuous optimization of traffic signals*, European Journal of Operational Research, 2007, vol. 181, pp. 1189-1197
26. Yperman I., *The Link Transmission Model for Dynamic Network Loading* (Ph. D. thesis)
27. Yperman I., Logghe S., Immers B., *The Link Transmission Model: an efficient implementation of the kinematic wave theory in traffic networks*, Advanced OR and AI Methods in Transportation