Network Signal Setting Design with Stage Sequence Optimisation

Silvio Memoli, Giulio E. Cantarella*, Stefano de Luca, Roberta Di Pace
Department of Civil Engineering, University of Salerno Italy, EU
smemoli@unisa.it; g.cantarella@unisa.it;sdeluca@unisa.it;rdipace@unisa.it;
* Corresponding author.

REVISED MANUSCRIPT PAPER 'TRB_2016_249'

Abstract

One of the most straightforward short term policies to mitigate urban traffic congestion is control through traffic lights at a single junction or network level. Existing approaches for *single junction Signal Setting Design* (SSD) can be grouped into two classes: Stage-based or Phase-based methods. Both these approaches take the lane marking layouts as exogenous inputs, but lane-based optimisation method may be found in literature, even though for isolated signal-controlled junctions only. The *Network Signal Setting Design* (NSSD) requires that offsets are introduced; a traffic flow model is also needed to compute total delay. All existing methods for NSSD follow a stage-based approach; these methods do not allow for stage matrix optimisation: it is shown that explicit enumeration of stage sequences is only practicable for very small networks.

This paper focuses on Network Signal Setting Design introducing the so-called *scheduled synchronisation* that includes green scheduling, green timing and coordination into one optimisation problem. The paper proposes a stage-based method to solve such a problem, as an extension of the synchronisation method and the traffic flow model proposed in Cantarella et al. (2015): first a set of candidate stages is defined for each junction, then the stage sequences, the stage lengths and the offsets are optimised all together. To the authors' knowledge, no other one-step optimisation method is available in literature for scheduled synchronisation. Results of the proposed method to a small network were compared with those from explicit enumeration of all stage sequences; results for a larger network are also discussed.

Keywords*: Network Signal Setting Design; Scheduled synchronisation; Stage Based methods; Macroscopic Traffic Flow modelling; Meta-heuristics.*

INTRODUCTION AND CONTRIBUTION

In order to mitigate urban traffic congestion, several policies can be adopted and may be applied in the short or long time horizon. With regard to the short term policies, one of the most straightforward is control through traffic lights at a single junction or network level.

With respect to this aim, existing approaches for *single junction Signal Setting Design* (SSD) - in off-line or planning applications - can be grouped into two classes:

- Stage-based (e.g. Webster, 1958; Allsop, 1971 and 1975; Burrow, 1987) methods,
- Phase-based (e.g. Improta and Cantarella, 1984; Gallivan, and Heydecker; Silcock, 1997; Wong, 1996, 1997; Wong et al., 2002) methods.

Stage-based signal setting methods divide the cycle into stages, each one being a time interval during which some mutually compatible approaches have green. Stage composition (say which approaches have green in each stage), and their sequence are traditionally represented through the so-called stage matrix. Once the stage matrix is given for each junction, the stage lengths (and possibly the cycle length) can be optimised with respect to several objective functions, such as minimization of total delay or maximisation of capacity (well-known methods of this kind are SIGSET and SIGCAP).

Phase-based methods address the signal setting as a periodic scheduling problem: the cycle length, the start and the length of the green period for each approach, and a binary variable for each pair of incompatible approaches are considered as optimisation variables see for instance the Binary Mixed Linear Program proposed by Improta and Cantarella (1987). In this case, the stage composition and sequence may easily be obtained from optimisation variables, thus, the stage matrix is an post-process result of the procedure. Some commercial software codes following this methodology are available, for instance OscadyPro®TRL (Burrow, 1987).

It is worth noting that all the above referred papers, both for the stage- based and the phase-based approaches, take the lane marking layouts as exogenous inputs, thus not included within the optimisation procedures proposed in this paper. On the other hand, developments on the lane-based optimisation method, for isolated signal-controlled junctions only, may be found in literature, from Lam et al. (1997) to more recently Wong et al. (2000), Wong and Wong, (2003), Wong et al. (2006); Zhou and Zhuang (2014), Sun et al. (2015), Yu et al. (2016).

The *Network Signal Setting Design* (NSSD) requires that further variables are introduced: the offsets that define when the signal setting plan of each junction starts with respect to a given clock reference. A traffic flow model is also required to compute total delay.

All existing methods for NSSD follow a stage-based approach. These methods do not allow for stage matrix optimisation thus, the effects of stage composition and sequence on network performance are not considered. It is worth noting that explicit enumeration of stage sequences is only practicable for very small networks (as shown in section 3).

For instance, TRANSYT15®TRLand TRANSYT-7F®FHWA allow to compute the cycle length, the green times and the offsets by combining a traffic flow model and a signal setting optimiser. In particular, TRANSYT15® generates several, but not all, significant stage sequences to be tested but the optimal solution is not endogenously generated, while TRANSYT-7F® is able to optimise the stage sequence for each single junction starting from the ring and barrier NEMA (i.e. National Electrical Manufacturers Association) phases. Still, these methods do not allow for complete stage matrix optimisation; moreover, the effects of stage composition and sequence on network performance are not analysed.

Other analyses may be also found in Hadi and Wallace (1993; 1994) which studied the possibility of introducing a phase sequence optimisation capability to TRANSYT-7F® using Genetic algorithms and the Cauchy simulated annealing. Moreover, with respect to the stage sequence optimisation Park et al. (2000) developed a simulation framework made up of a mesoscopic flow simulator and Genetic Algorithm optimiser, and showed that the simulator provided better results than those obtained using the software TRANSYT-7F®.

More recent contributions based on binary variables have been proposed by Friesz et al. (2013), Liu and Smith (2015), Smith et al. (2015) for switch-on / switch-off signals in the case of within-day dynamic traffic assignment; some authors have introduced the continuum signalised junction model (Han et al., 2014; Han and Gayah, 2015) to avoid discrete variables.

The existing phase-based methods are available for single junctions only. Currently, in practical applications, once the green timing and scheduling have been carried out for each junction through a phase-based method, offsets can be optimised (coordination) using the stage matrices and greens obtained from single junction optimisation or using the stage matrices only to optimise both greens and offsets (synchronisation), through a stage-based method, as those above described.

This paper focuses on Network Signal Setting Design (NSSD) introducing the so-called *scheduled synchronisation* that includes green scheduling, green timing and coordination into one optimisation problem. The paper proposes a stage-based method to solve such a problem (see section 2.). In this method, first a set of candidate stages is defined for each junction, then the stage sequences, the stage lengths and the offsets are optimised all together. To the authors' knowledge, no other one-step optimisation method is available in literature for scheduled synchronisation

The proposed method, called CENEO (ComplEte NEtwork Optimisation), is an extension of the synchronisation method and the traffic flow model proposed in Cantarella et al. (2015) that requires the stage sequence as input data and optimises stage lengths and offsets only (called ENEO in the rest of the paper for easy reference).

As already said, existing phase-based methods are not available for networks of junctions. These methods might be formally extended to specify one-step methods for NSSD (offsets are quickly obtained the start and the length of the green period of each approach). Nonetheless, the resulting Mixed Optimization Program is hard to solve since several equivalent local optima exist; this condition may quite easily be dealt with for a single junction, but it is still unclear how it can effectively be circumvented for a network (with loops). So far stage-based methods for single junctions can more simply be extended to scheduled synchronisation by explicitly including the stage sequencing within the optimisation method, after the stage generation step, as said above. Resulting methods are simpler to specify than those derived from phase-based methods. These reasons and others discussed in subsection 2.4 support the stage-based approach pursued in this paper.

The paper is organised as follows: section 2 describes the proposed method CENEO, and some consideration about advantages with respect to phase-based methods are also discussed; in section 3 the results of the numerical applications for two toy networks carried out through CENEO are shown; in section 4 the final conclusions are discussed. A list of notations is included at the end of the paper.

1 PROPOSED METHOD

In this section the proposed approach to the scheduled synchronisation is discussed.
As already noted in the introduction, stage based methods can be used to solve the scheduled synchronisation by explicitly including the stage sequencing within the optimisation method, after the stage generation as described below.

1.1 STAGE DEFINITION AND GENERATION

A *stage* is a set of *approaches* that have green at the same time. For safe operations all the approaches in a stage must be mutually compatible, namely they may have green without any conflict– *compatibility requirement*–. Usually it is also required that each stage be maximal – *completeness requirement* –, meaning that no further approach may be added to any of them without violating the compatibility requirement.
All candidate stages satisfying both the compatibility and completeness requirements can easily be generated by Bron & Kerbosch's (1973) algorithm for finding all maximal cliques of a graph; in this case, the adjacency matrix of the graph is the (square symmetric) compatibility matrix with as many rows and columns as the number of approaches and '0/1' entry for each 'incompatible/compatible' pairs of approaches.
Let $n \geq 2$ be the number of candidate stages ($n = 1$ meaning there is no pair of incompatible approaches, there is no need for traffic control). A candidate stage that contains an approach not included in any other stage is called *compulsory*, otherwise it is called *optional*. Let n_c and $n_o = n - n_c$ be the number of compulsory and optional candidate stages, respectively.

1.2 VARIABLES AND CONSTRAINTS

In this sub-section main variables and constraints among them are described to build up the optimisation model described in sub-section 2.4.

2.2.a Stage sequences

A set of candidate stages (not necessarily all of them) in a given order are a *sequence*. A sequence is called *feasible* if each approach belongs to at least one stage in the sequence – *feasibility requirement* –. If all the compatible and maximal stages are considered such condition surely holds (in the limit case each stage contains only one approach). Moreover, a compulsory candidate stage, as defined above, must be included in each

sequence since it contains an approach not included in any other stage. The number of feasible sequences may be very large.

If there is no optional stage, n_o= 0 and $n = n_c$, the number of feasible sequences is given by n!, say the number of permutations of the n stages. However, it is worth noting that any periodic rotation of a sequence, for instance such that the sequence (1,2,3) becomes (2,3,1), then (3,1,2), then (1,2,3,) again, does not affect optimal green and performance indicators, while optimal offsets change in an easily predictable way–(cfr sub-section 2.2 and example in Figure 1 in the following).

Thus, for each junction all the sequences are considered grouped into (n_i! / n_i) = (n_i– 1)! equivalence classes, and only one sequence for each equivalence class is further analysed. Thus,

- if only two stages are available, two possible sequences can be built up {(1,2); (2,1)} which are equivalent, thus, only one equivalent class exists;
- if three stages are available two equivalent classes exist: {(1,2,3), (2,3,1); (3,1,2)} and {(3,2,1); (2,1,3); (1,3,2)}; either may be obtained from sequence (1,2) by positioning stage 3 after stage 1 or after stage 2; thus a binary variable is enough to define the optimal sequence;
- if four stages are available six equivalent classes exist: {(1,2,3,4); (2,3,4,1); (3,4,1,2); (4,1,2,3)}; {(1,3,2,4); (3,2,4,1); (2,4,1,3); (4,1,3,2)}; {(1,4,2,3); (4,2,3,1); (2,3,1,4); (3,1,4,2)}); {(1,4,3,2); (4,3,2,1); (3,2,1,4); (2,1,4,3)}; {(1,3,4,2); (3,4,2,1); (4,2,1,3); (2,1,3,4)}; {(1,2,4,3); (2,4,3,1); (4,3,1,2); (3,1,2,4)}); each may be obtained by changing the position of stage 4 with respect to sequences (1,2,3) or (3,2,1); thus a binary variable and a ternary variables are enough to define the optimal sequence;
- and so on.

If there is at least one optional stage, n_o> 0, n>n_c, the stages can be grouped into 2^{n_o} sub-sets, each including all the n_c compulsory stages and some (or none at all) of the n_o optional stages; stages belonging to any of such sets may be arranged in a number of feasible sequences equal to the factorial of its size minus one, as described above. Thus,

- if only one compulsory, 1, and one optional, 2, stages are available it implies that stage 2 is a sub-set of stage 1, therefore, stage 2 violates the completeness requirement, even though two equivalent classes exist: {(1)}; {(1,2); (2,1)}, thus this case it is not further considered;
- if two compulsory, 1 and 2, and one optional, 3, stages are available, three equivalent

classes exist: {(1,2); (2,1)}; {(1,2,3), (2,3,1); (3,1,2)} and{(3,2,1); (2,1,3); (1,3,2)}; each may be obtained from sequence (1,2) by not including stage 3 or by positioning it after stage 1 or after stage 2; thus a ternary variable is enough to define the optimal sequence;

- if three compulsory, 1 and 2 and 3, and one optional,4, stages are available, eight equivalent classes exists: {(1,2,3), (2,3,1); (3,1,2)} and {(3,2,1); (2,1,3); (1,3,2)}; {(1,2,3,4); (2,3,4,1); (3,4,1,2); (4,1,2,3)}; {(1,3,2,4); (3,2,4,1); (2,4,1,3); (4,1,3,2)}; {(1,4,2,3); (4,2,3,1); (2,3,1,4); (3,1,4,2)}); {(1,4,3,2); (4,3,2,1); (3,2,1,4); (2,1,4,3)}; {(1,3,4,2); (3,4,2,1); (4,2,1,3); (2,1,3,4)}; {(1,2,4,3); (2,4,3,1); (4,3,1,2); (3,1,2,4)}; each may be obtained by not including stage 4 or by changing its position within the sequences (1,2,3); thus three binary variables are enough to define the optimal sequence;
- and so on.

Quite often it is also required that each approach has green in consecutive stages within the sequence, if more than one – *consecutiveness requirement* –.This requirement is effective only for sequences containing four or more stages.

The sequence of stages and their composition is commonly described by Δ, the approach-stage incidence matrix (or stage matrix for short), with entries δ_{kj}=1 if approach *k* receives green during stage *j* and 0 otherwise.

2.2b Time and flow variables

Let the junction network be represented by an undirected graph with a node for each junction and an edge for each pair of adjacent junctions (the actual traffic directions are irrelevant), with *m* nodes as the number of junctions.Assuming that the green scheduling is described by the stage composition and their sequence as discussed above, let

$c > 0$ be the cycle length, common to all junctions, assumed known or as a optimisation variable, often constrained in a given range.

For each junction (not explicitly indicated), after Webster (1958), let

$t_j \in [0, c]$ be the length of stage *j* as a optimisation variable; if no minimum length constraint is introduced, the optimal length of a (optional) stage may be zero, meaning that this stage is not in the optimal solution;

$t_{ar} \in [0, c]$ be the so-called all red period at the end of each stage to allow for the safe clearance of the junction, assumed known (and constant for simplicity's sake);

$l_k \in [0, c]$ be the lost time for approach *k*, assumed known;

$g_k = \sum_j \delta_{kj} t_j - t_{ar} - l_k \in [0, c]$[1] be the effective green for approach *k* needed for computing total delay through a traffic flow model (see next sub-section 2.3).;

$q_k > 0$ be the arrival flow for approach *k*, assumed known;

$s_k > 0$ be the saturation flow for approach *k*, assumed known.

Apart from non-negativity of optimisation variables, a constraint needs to be introduced for each junction in order to guarantee consistency among the stage lengths and the cycle length:

$$\sum_j t_j = c$$

Other constraints are sometimes used in order to guarantee

the minimum value of the effective green

$$g_k \geq g_{min} \qquad \forall_k$$

and that the capacity factor is greater than 1 (or any other value assumed as a threshold)

$$\left(\frac{s_k \cdot g_k}{c \cdot q_k}\right) \geq 1$$

Such a constraint may be added only after having checked that the maximum junction capacity factor for each approach *k* in the junction *i* is greater than 1, otherwise, a solution may not exist whatever the objective function is.

Moreover, for each junction *i* let

$\phi_i \in [0, c[$ be the node offset between the start of a reference stage of junction *i* and the start of the reference stage of the first junction used as a reference for clock, $\phi_1 = 0$, thus the number of the independent node offsets is *m* - 1.

For each pair of (adjacent) junctions (*i, h*) in the network, let

$\phi_{ih} = \phi_h - \phi_i = -\phi_{hi}$ be the link offset between junctions *i* and *h*, needed for computing total delay through a traffic flow model (see next sub-section 2.3).

If the network contains *k* > 0 independent loops, the number of independent link offsets is equal to *m-k,* thus it is better to use the *m* - 1 independent node offsets as optimisation variables to avoid an useless increase of the number of variables and of constraints. On

---

[1] non-negative effective green is usually guaranteed by the non-negative stage length, but for a very a short cycle length with regard to the values of all-red period length and lost times, say the cycle length is less than $\sum_j MAX_k (\delta_{kj} l_k + t_{ar}) \geq c$, this condition is never met in practical applications.

the other hand, if the network is loop-less, all the *m* - 1 link offsets are independent, as many as the independent node offsets, and might be used as optimisation variables instead of the independent node offsets; arterials are a special case of such kind of networks.

As noted above, after any periodic rotation of a sequence, for instance such that the sequence (1,2,3) becomes (2,3,1), then (3,1,2), then (1,2,3,) again, optimal green and performance indicators remain unchanged, while optimal offsets change in an easily predictable way, thus all those sequences have been called equivalent, making up an equivalence class. For instance, the layout in Figure 1 refers to a simple arterial in which the network cycle length is equal to 60 seconds, the length of stages for the upstream junction are 28 seconds, 17 seconds and 15 seconds whilst the length of the stages for the downstream junction are 40 seconds and 20 seconds; three combination of sequences are evaluated: i) in case of sequences 123-12, ϕ_{11}= 50 s, ϕ_{21}= 50 – 28 = 22 s,ϕ_{31}= 50 – (28+17) =5 s*;* ii) in case of sequences 231-12, ϕ_{21}= 22 s, ϕ_{31}= 22 -17=5 s,ϕ_{11}= 22 – (17+15) = - 10 + 60 = 50 s*;* iii) in case of sequences 321-12, ϕ_{31}= 5 s, ϕ_{21}= 5 -15=-10+60=50 s,ϕ_{11}= 5 – (15+17) = - 27 + 60 = 33 s. This examples shows that sequence 123 is equivalent with sequence 231 whilst is not equivalent with sequence 321.

- place Fig.1 # about here

1.3 OBJECTIVE FUNCTIONS

In this paper the total delay TD was considered as objective function (even though other measure of performances could be introduced). Since generally in a junction network there can be interacting and non-interacting approaches, two cases should be distinguished.

DELAY FOR NON-INTERACTING APPROACHES

In this case it was computed using the two term Webster formula (Webster, 1958) as

$$TD^j_k = \sum_k 0.45 \cdot q^j_k \cdot c_j \cdot \left(1 - g^j_k/c_j \right)^2 / \left(1 - q^j_k / s^j_k \right) + \\ + 0.45 \cdot q^j_k / \left(\left(s^j_k \cdot g^j_k/c_j \right) \cdot \left(\left(g^j_k/c_j \right) \cdot \left(s^j_k / q^j_k \right) - 1 \right) \right) \qquad (1)$$

Any other formula can be used as well.

DELAY FOR INTERACTING APPROACHES

In this case delay computation needs an explicit *traffic flow model*. In this paper the model described in Cantarella et al. (2015), was adopted, but any other traffic flow model can be used as well. For the computation of delay for an interacting approach the cumulated input, $Ciq_k^j(t)$, and output flows, $Coq_k^j(t)$, through the stop line of approach *k* of junction *j*, in the subsequent sub-intervals *t* were compared.

The Deterministic Total Delay (DTD_k^j) cumulated in the interval $[0,T_a]$ for approach *k* of junction *j* was then given by the following expression:

$$DTD_k^j = \sum_{t=1..\,T_a/\Delta\tau} (Ciq_k^j(t \times \Delta\tau) - Coq_k^j(t \times \Delta\tau))\, \Delta\tau \quad [2] \qquad (2)$$

Thus delay experienced at an interacting approach is a function of the offsets between the timing plans. In fact, such a delay depends on the output flow in the downstream junction which is obtained by starting from the input flow in the upstream junction through the phenomenon of dispersion.

Let s_k^j be the saturation flow on approach *k* of junction *j*, the Stochastic and Oversaturation component of Total Delay $SOTD_k^j$ for approach *k* of junction *j* is computed using the following expression

$$SOTD_k^j = \{ [(q'^j_k \cdot s^j_k)^2 + (4q'^j_k / T_a)]^{0.5} + (q'^j_k \cdot s^j_k) \}\, T_a / 4 \qquad (3)$$

and considering the average of the values of the cyclic flow profile along the connecting link arriving at approach *k* of considered junction *j*, q'^j_k as input flow.

Then

$$TD_k^j = DTD_k^j + SOTD_k^j \qquad (4)$$

Finally, the total network delay is given by

$$TD = \sum_j \sum_k TD_k^j \cdot \qquad (5)$$

where TD_k^j is computed through equation (1) in case of non-interacting junction and through equation (4) in case of interacting junctions.

[2] $\Delta\tau$ the sub-interval of equal duration in which the time of analysis T_a is divided.

1.4 OPTIMISATION MODEL

A mixed discrete-continuous linear optimisation problem with non-linear objective function (not available in closed form for interacting approaches) is obtained by combining together:

- the discrete variables needed to define the optimal stage sequence, constrained by the feasibility and the consecutiveness requirements, as in sub-section 2.2.a;
- the continuous variables needed to completely define the signal plan, that is: (i) the stage lengths, as many as the stages, constrained by the consistency among the stage lengths and the cycle length, (ii) the m - 1 (independent) node offsets, and possibly (iii) the cycle length, as well as other constraints introduced in sub-section 2.2.b;
- the objective function defined by the total delay, as in sub-section 2.3.

The number and/or the kind of discrete variables may be reduced if no minimum length constraint is introduced, in this case indeed if the optimal length of a (optional) stage is zero it means that this stage is not in the optimal solution.
By a comparison, extending to a junction network for instance the phase-based method proposed by Improta and Cantarella (1984) for as single junction would require two continuous variables for each approach and one binary variable for each pair of incompatible approaches, thus leading to a much larger number of optimisation variables.

1.5 OPTIMISATION ALGORITHM

To solve optimisation problems of the kind described in the previous sub-section, meta-heuristic algorithms are usually adopted such as Simulated Annealing (SA). As a matter of fact, such algorithms can effectively address even optimisation problems with objective function not expressed in closed form, so that derivatives are not easily available, as it occurs for the scheduled synchronisation. The implementation remarks of CENEO are outlined below.

The proposed SA[3] algorithm mimics the annealing process of a metal, also known as the Metropolis Scheme (Metropolis et al., 1953), starts with an initial solution and goes through

[3]SA can deal with non-linear models, unordered data with many constraints. Its main advantages over other local search methods are its flexibility and its ability to approach global optimality.

a predetermined number of iterations to try to improve upon the objective function. In the scheduled-synchronisation, the algorithm starts with an initial combination of sequences, greens and offsets, randomly provided, and tries to improve the network total delay (or any other measure of performance). The whole framework is composed by an outer loop (external iterations) and an inner loop (internal iterations), as in following described (see Figure 2).

- place Fig.2 # about here

OUTER LOOP

From the initial or current solution (combination of sequences, greens and offsets), and an initial value of the so-called current temperature of the system *T*, at each external iteration, a new solution is randomly generated from the neighbourhood according to a generation scheme (see below for the inner loop). The measure of performance (to be minimised) of the current new solution, c.n, is calculated and compared with the current optimum one, c.o .

If the new measure, c.n, is better than the current optimum one, c.o, $(c.n - c.o) \leq 0$, it is accepted anyhow (say, with a probability of one) and the new solution becomes the current optimum one, then the process repeats itself. However, if the new measure, c.n, is worse than the current optimum one, c.o, $(c.n - c.o) > 0$, then the new solution may be still accepted but with a probability $P(c) = \exp(-(c.n - c.o) / T) < 1$; in particular, given a real number (*r*) randomly drawn from a uniform distribution over the interval [0,1], if P(c) is smaller than the number *r*, then the new solution is rejected and the current solution remains in the process while the algorithm repeats itself.

The temperature (*T*) is decreased at each iteration by a predetermined percentage until it reaches a certain predetermined value where the SA process terminates. To this aim it is required the specification of an annealing schedule, i.e. an initial temperature (T_{init}) and the rules for lowering it as the search progresses until a final Temperature (T_{fin}). Several cooling schedules are covered in literature, including exponential, linear and temperature cycling. In this paper we adopt the Kirkpatrick geometric cooling scheme, considering at each iteration *it* a temperature $T_{it} = \alpha T_{it-1}$, where α is a constant close to, but smaller than 1, usually assumed equal to 0.95.

Note that as temperature decreases, so decreases the probability of accepting a “hill-climbing move.” If *T* is very large, then *r* is likely to be less than P(c) and the new solution is almost always accepted. If *T* is small, or close to zero, then only if the new solution is characterised by a very small $\Delta c > 0$ it has any realistic chance of being accepted.

INNER LOOP

At each external iteration (temperature state), the new solution is carried out as an output of an inner loop which consists of testing the greens and offsets, *x*, randomly provided for the current external iteration, for a number (*N*) of combination of sequences, *s*, (equalling the number of internal iterations of the inner loop) extracted within the set of sequences representing the $(n_i - 1)!$ equivalence classes of each junction. The size of the set of combinations to be tested is determined by the number of junctions in the considered network. For instance, in the case of a three junction network, with junction 1 having 3 stages and 2 equivalence classes, junction 2 having 4 stages and 6 equivalence classes, junction 3 having 3 stages and 2 equivalence classes, the size of such a set is given by $2 \times 6 \times 2 = 24$ combinations of sequences.

The generation scheme is carried out on discrete binary and tertiary variables; in particular, if the considered junction contains a sequence of three stages, two equivalence classes exist thus a binary (b) variable 0/1 is adopted to randomly choose between the sequence (1,2,3), belonging the class {(1,2,3), (2,3,1); (3,1,2)} and the sequence (1,3,2), belonging the class {(3,2,1); (2,1,3); (1,3,2)}. On the other hand, if the considered sequence contains four stages, a binary (b) variable 0/1 and a ternary (t) variable 1/2/3 is considered and two random generations are applied (see Figure 3): first a binary random decision is taken as to select between the sequence (1,2,3,4), belonging the class {(1,2,3,4); (2,3,4,1); (3,4,1,2); (4,1,2,3)}, and the sequence (1,4,3,2), belonging the class {(1,4,3,2); (4,3,2,1); (3,2,1,4); (2,1,4,3)}, then a ternary random decision is taken as to select the positions of the last stage in the sequence: (1,4,2,3) xor (1,2,4,3) xor (1,2,3,4), otherwise (1,4,3,2) xor (1,3,4,2) xor (1,3,2,4). And so on for junctions with more than four stages.

It is quite clear that the generation scheme is constrained by the requirements described in sub-section 2.2. Necessarily the chosen sequences had to respect the feasibility and consecutiveness requirements. That implies, in some cases, discarding solutions in which some approaches had green in two non-consecutive stages; this way the number of stage sequence combinations to be test is further reduced.

- place Fig.3 # about here

In order to strengthen the algorithm exploitation, at each external iteration, the current optimum sequence combination, *s*, is stored and fixed for the successive neighbourhood

generation steps (in this way the neighbourhood size will be equal to the total number of feasible combinations minus one) for the purpose of intensifying (refining) the search of the optimal greens and offsets, *x*, in the vicinity of *s*.

For readers ‘convenience an example will be shown in the following Table 9.

2 NUMERICAL APPLICATIONS

In order to test the proposed method CENEO, implemented by the authors in a MATLAB (Release 2013b) code, two toy networks, case study A and B, were considered as described in the following. In particular case study A was introduced in order to

1. point out the relevance of the green scheduling on the network performances;
2. compare the results of CENEO with those obtained by explicitly enumerating all possible stage sequences.

An extensive application of CENEO providing further details regarding the optimisation procedure is shown in case study B.

2.1 CASE STUDY (A)

In this section the scheduled synchronisation problem is applied to a small size triangular network. Both the explicit enumeration and the implicit enumeration stage based method were performed; the solution results are then compared. For the sake of clarity we need to state that the two solution approaches are not completely comparable due to the fact that in the stage based implicit enumeration method, the sequences are simultaneously optimised together with the greens and the offsets whilst in the explicit enumerative method two separate steps are identified: 1) the selection of the feasible combination of sequences; 2) the optimisation of the greens together with the offsets which are constrained to the selected combination of sequences.

It has to be noted that in the following case study we will indicate, with the *number of external iterations* (*ei*), the number of temperature states adopted to develop the simulated annealing scheme, while with the *number of internal iterations* (*ii*) the number of combinations (randomly selected from the equivalence classes sets) tested within each external iteration simultaneously with the current greens and offsets.

On the basis of previous considerations it is easy to understand how in the case of the explicit enumerative approach a higher number of traffic flow simulations[4] is required, if

[4] The number of traffic flow simulations in case of explicit enumerative approach is equal to the product of the size of the set of feasible combinations of sequences and the number of algorithm iterations during the second step of the procedure, i.e. for the synchronisation.

compared to the implicit approach[5]. Nevertheless, such a comparison is performed to investigate the effectiveness of the proposed implicit method.

The test network layout is shown in Figure 4, the entry/exit flows and the turning percentages (obtained through path flows) are shown respectively in Table 1 and Table 2; path-flow patterns are summarised in Table 3; finally, in Figure 5 the stage compositions and the equivalence classes of sequences are summarised. In accordance with the previous section which explains the optimisation problem, let m be the number of junctions in the network equal to 3 and n_i the number of feasible stages for each junction i (equal to 3 for junction 1, 4 for junction 2 and 3 for junction 3), and let us assume that all stages are compulsory for simplicity's sake, there are $(n_i - 1)!$ classes of equivalent sequences for each junction i (i.e. 2 classes for junction 1, 6 classes for junction 2 and 2 classes for junction3), thus, there are $\prod_{i=1,\ m} (n_i - 1)!$ combinations of classes, equal to 24, to be analysed in order to establish the optimal one. Moreover, in our implementations two equivalent classes were further discarded at junction 2 (i.e. classes 1324 and 1423) since the approach j_{22} had green in two non-consecutive stages, so reducing the number of stage sequence combinations to 16.

- place Fig.4 # about here

As shown in the following of this section, the outputs of the proposed method, CENEO, even testing the significance of the number of the internal iterations upon the resulting performance (in terms of time consuming and network total delay minimisation) are analysed.

Then a comparison with explicit enumeration, yield through ENEO, has been carried out. It is worth remembering, as regards ENEO implementations, how the number of algorithm iterations can affect the overall implementation (substantially in terms of time consuming since related to the number of traffic simulations to be run). On the base of such a consideration only the case in which ii = 5 will be discussed and then compared with the outputs of CENEO.

[5] The number of traffic flow simulations in case of implicit approach is equal to the product of the *ii* which is always lower than the size of the complete set of combinations and the *ei* which is fixed as in the explicit approach.

Table 1: entry-exit flows

entry/exit	x1	x2	x3	x4	Total
x1	-	179	170	225	574
x2	132	-	45	39	216
x3	143	32	-	73	248
x4	302	69	28	-	399
Total	577	280	243	337	1437

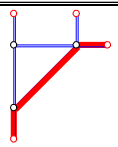
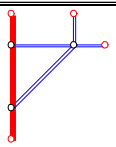
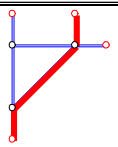
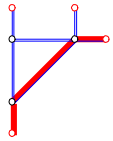
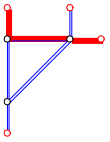
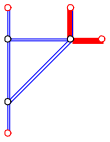
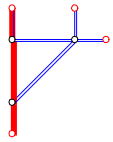
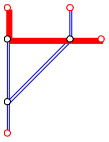
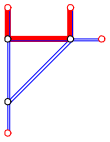
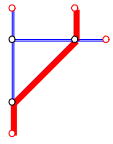
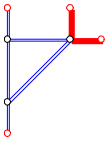
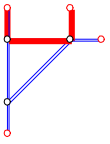
Table 2: turning percentages

Source	turn [%]		
link	Ahead	Left	Right
x1-1	*	-	*
2-1	60	40	-
3-1	-	70	30
x2-3	70	30	*
x4-3	60	10	30
1-3	30	30	40
2-3	20	40	40
1-2	70	-	30
3-2	-	70	30
x3-2	*	*	-

*exclusive turn lane

- place Fig.5 # about here

Table 3: traffic flow patterns

to / from	x1	x2	x3	x4
x1				
x2				
x3				
x4				

As previously described, the first evaluation of the CENEO modelling has concerned its performances both in terms of effectiveness (total delay minimisation) and in terms of efficiency (running time). As Table 4 makes clear, no meaningful distances among the optimisations outputs (TD) arise by adopting a different number of internal iterations, *ii*. That is presumably due to the low complexity of the network which implies that good approximation of the optimal global solution can be reached also with a low complexity of the selected algorithm iteration. On the other hand, as expected, different running times could be pursued; although, considering that this discrepancy is measured in terms of minutes it could be stated that, since a certain number of internal iterations *ii*, no relevant advantages in terms of overall performance could be reached by sliding such a parameter.

Table 4: performance indicators w.r.t. different values of internal iterations (ii=3, 5 and 8)

internal iterations	running time [min]	TD [PCU h/h]
3	7	22.74
5	8	22.06
8	10	21.94

As stated before, the comparison has been led by considering 5 internal algorithm iterations (*ii* = 5). Starting from CENEO implementations, several runs[6] are performed in order to verify the internal dispersion of solution results which was affected by the stochastic nature of the optimisation algorithm. In Table 5 and Table 6 only the results of the first shot are shown; specifically, in Table 5 for each junction the optimum sequence is displayed whilst Table 6 shows the stage durations, t_j, and the link offsets, ϕ_i, with $\phi_1 = 0$. In Table 7 and Table 8 the dispersion among different runs is outlined. As expected, a small internal dispersion has been reached and in particular the first, the second and the third minimum are very close. Moreover, the fluctuations are less than the 9% of the best solution.

[6] In general there is no indication about the number of runs to be carried out. In our case it was observed that the optimal solution was not significantly affected if the considered number of runs was greater than 8.

Table 5: optimal sequences computed through one shot CENEO

CENEO			
junction 1	junction 2	junction 3	TD
optimum sequence	Optimum sequence	Optimum sequence	[PCU-h/h]
123	1432	123	22.06

Table 6: optimal greens and offsets computed through one shot CENEO

CENEO												
junction 1			junction 2					junction 3				TD
t_1	t_2	t_3	t_1	t_2	t_3	t_4	ϕ_2	t_1	t_2	t_3	ϕ_3	[PCU-h/h]
[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	
22	42	36	16	37	19	28	48	25	36	39	8	22.06

Table 7: optimal sequences computed through CENEO over several runs

CENEO					
junction 1	junction 2	junction 3	TD	Sim run	Delta to best
Optimum sequence	Optimum sequence	Optimum sequence	[PCU-h/h]		[%] Internal dispersion
123	**1432**	**123**	22.06	1	+3.04
123	**1234**	**132**	21.41	2	**0.00**
123	**1432**	**132**	23.32	3	+8.92
123	**1432**	**132**	23.07	4	+7.75
123	**1432**	**123**	23.33	5	+8.97
123	**1234**	**132**	21.58	6	+0.79
132	**1234**	**132**	22.47	7	+4.95
123	**1234**	**132**	21.63	8	+1.03

Table 8: optimal greens and offsets computed through CENEO over several runs

CENEO														
junction 1			junction 2					junction 3				TD	Sim run	Delta to best
t_1	t_2	t_3	t_1	t_2	t_3	t_4	ϕ_2	t_1	t_2	t_3	ϕ_3	[PCU-h/h]		[%]
[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]	[s]			Internal dispersion
22	42	36	16	37	19	28	48	25	36	39	60	22.06	1	+3.04
24	44	32	16	32	21	31	57	27	33	40	65	21.41	2	**0.00**
23	48	29	15	35	18	32	88	27	36	37	11	23.32	3	+8.92
27	44	29	13	28	24	35	39	29	34	37	11	23.07	4	+7.75
25	43	32	17	34	16	33	62	25	31	44	42	23.33	5	+8.97
20	47	33	20	31	20	29	64	28	35	37	38	21.58	6	+0.79
23	40	37	14	37	16	33	19	27	38	35	79	22.47	7	+4.95
25	41	34	13	29	24	34	13	26	33	41	65	21.63	8	+1.03

In the following Table 9 an excerption of the neighborhood dynamic generation in a SA execution is shown. In particular, internal redundancy of the random solution generated during each SA iteration is tested in order to verify the algorithm exploration capability which consists of probing a much larger portion of the search space with the hope of finding other promising solutions that are yet to be refined. Furthermore, total redundancy is carried out in order to test algorithm exploitation, intensifying and refining the search in the vicinity of the current optimum sequence combination.

Table 9: dynamic neighbourhood generation for CENEO (an excerption)

SA iteration	Neighbourhood generation		Junction 1	Junction 2	Junction 3	Internal redundancy #/5	Total redundancy #/120
	1		123	1234	123	1	7
	2		123	1243	132	1	9
1	**3**		123	1432	123	1	12
	4	*	123	1243	123	1	8
	5		132	1234	123	1	11
	1		123	1243	123	1	8
	2		132	1432	123	1	8
2	**3**		132	1342	132	2	5
	4		132	1342	132	2	5
	5	*	123	1234	123	1	7
	1		123	1234	123	1	7
	2		123	1342	132	1	10
3	**3**	*	132	1243	132	1	11
	4		123	1243	132	1	9
	5		132	1432	132	1	6
	~		~	~	~	~	~
	1	*	123	1432	123	2	12
	2		132	1432	132	1	6
23	**3**		123	1234	132	1	4
	4		132	1243	132	1	11
	5		123	1432	123	2	12
	1	*	123	1432	123	1	12
	2		123	1342	132	1	10
24	**3**		132	1342	123	1	4
	4		132	1243	132	2	11
	5		132	1243	132	2	11

* current optimum (c.o)

The results obtained through CENEO have been then compared with those achieved by explicitly enumerating and analysing all feasible sequences through the procedure ENEO. This latter approach is clearly suitable only for a very small simple network (as occurred in this case study) due to the large number of feasible sequences greatly increasing with the number of junctions and the number of stages (see case study B) and it is presented below for comparison purpose only.

In the ENEO procedure, three implementation steps are identified: 1) for each junction the adjacency (or compatibility) matrix is defined then all the equivalence classes of sequences are generated; 2) all the $\prod_{i=1,\ m} (n_i - 1)!$ combinations of sequences (of all the network junctions) are created; 3) for each combination, greens and offsets optimising network TD are computed simultaneously (synchronisation) through a Simulated Annealing solution algorithm (see Cantarella et al., 2015).In Table 10 and Table 11 the results obtained through ENEO are shown. As expected the combination of sequences greatly affect the network performances (the worst combination leads to TD close to 50% greater than the best combination) and, thus, support a proper investigation on the problem solution.

Table 10: optimal sequences computed through ENEO

ENEO				
junction 1	junction 2	junction 3	TD [PCU-h/h]	Delta to best [%] External dispersion
Enumerative sequence	Enumerative sequence	Enumerative sequence		
132	1234	132	22.01	+3.07
132	1243	132	23.14	+8.75
132	1342	132	22.93	+7.78
132	1432	132	27.46	+28.89
123	1234	132	21.57	+1.18
123	1243	132	24.85	+16.54
123	1342	132	26.82	+25.09
123	1432	132	21.43	**0.00**
123	1234	123	22.70	+6.23
123	1243	123	24.53	+14.67
123	1342	123	25.95	+21.45
123	1432	123	22.01	+3.14
132	1234	123	21.97	+3.08
132	1243	123	23.72	+11.27
132	1342	123	31.44	+47.11
132	1432	123	26.88	+25.00

Table 11: optimal greens and offsets computed through ENEO

ENEO													
junction 1			junction 2					junction 3				TD [PCU-h/h]	Delta to best [%] External dispersion
t_1 [s]	t_2 [s]	t_3 [s]	t_1 [s]	t_2 [s]	t_3 [s]	t_4 [s]	ϕ_2 [s]	t_1 [s]	t_2 [s]	t_3 [s]	ϕ_3 [s]		
28	40	32	13	31	20	36	86	29	31	40	42	22.01	+3.07
25	41	34	20	30	21	29	23	24	34	42	60	23.14	+8.75
25	43	32	23	29	23	25	29	27	30	41	15	22.93	+7.78
19	43	38	23	24	17	36	85	34	26	40	16	27.46	+28.89
22	44	34	15	38	18	29	30	25	38	37	26	21.57	+1.18
23	49	28	21	33	24	22	56	29	30	41	44	24.85	+16.54
19	46	35	20	32	18	30	25	22	40	38	22	26.82	+25.09
23	39	38	23	29	17	31	51	32	30	38	16	21.43	**0.00**
25	45	30	18	25	21	36	81	34	26	40	29	22.70	+6.23
27	47	26	19	34	24	23	13	27	29	44	66	24.53	+14.67
17	46	37	21	38	17	24	57	28	34	38	65	25.95	+21.45
25	41	34	19	25	18	38	94	28	27	45	56	22.01	+3.14
26	39	35	14	28	24	34	6	25	33	42	46	21.97	+3.08
26	40	34	21	26	24	29	94	28	30	42	94	23.72	+11.27
16	28	56	27	25	22	26	46	25	30	45	1	31.44	+47.11
29	45	26	20	33	18	29	32	30	26	44	44	26.88	+25.00

In the following Table 12 it is shown the internal dispersion of the solutions carried out through ENEO by comparing the results obtained in 8 simulation runs in which the combination of sequences is the best one previously obtained (see Table 9, sequence 123 for junction 1, 1432 for junction 2, 132 for junction 3, which lead to a TD of 21.43 PCU-h/h). It should be stressed that 2 solutions with equal sequence combination may still differ with respect to greens and offsets due to the stochastic nature of the algorithm adopted for approximating the global optimum (see Table 13). Anyhow the fluctuations are less than 6% of the best solution. It is worth noting that, as expected, the internal dispersion obtained in 8 simulation runs of CENEO (Table 7) results higher with respect to the internal dispersion in 8 simulation runs of ENEO (Table 12).

Table 12: optimal sequences computed through ENEO over several runs on the best combination of sequences

ENEO					
junction 1	junction 2	junction 3	TD [PCU-h/h]	Sim run	Delta to best [%]
Constrained sequence	Constrained sequence	Constrained sequence			Internal dispersion
123	1432	132	21.43	1	+2.19
123	1432	132	22.13	2	+5.53
123	1432	132	20.97	3	**0.00**
123	1432	132	21.33	4	+1.72
123	1432	132	21.49	5	+2.50
123	1432	132	21.41	6	+2.10
123	1432	132	22.05	7	+5.15
123	1432	132	21.37	8	+1.91

Table 13 optimal greens and offsets computed through ENEO over several runs on the best combination of sequences

ENEO														
junction 1			junction 2					junction 3				TD [PCU-h/h]	Sim run	Delta to best [%]
t_1 [s]	t_2 [s]	t_3 [s]	t_1 [s]	t_2 [s]	t_3 [s]	t_4 [s]	ϕ_2 [s]	t_1 [s]	t_2 [s]	t_3 [s]	ϕ_3 [s]			Internal dispersion
23	39	38	23	29	17	31	51	32	30	38	46	21.43	1	+2.19
25	41	34	17	30	23	30	35	29	33	38	27	22.13	2	+5.53
23	43	34	18	29	23	30	57	28	32	40	44	20.97	3	**0.00**
26	43	31	21	26	19	34	46	30	34	36	25	21.33	4	+1.72
23	47	30	16	30	23	31	55	26	34	40	48	21.49	5	+2.50
23	42	35	19	31	21	29	60	26	34	40	63	21.41	6	+2.10
24	42	34	18	29	21	32	72	27	33	40	62	22.05	7	+5.15
22	45	33	19	29	19	33	61	26	34	40	53	21.37	8	+1.91

The analysis of Table 12 and Table 13 might be repeated for all, or at least for the best solutions in Table 9 (for example for the solutions that return TD not higher than 6% of the best one).

The final step of the application shown in this section concerned the comparison among the two methods (explicit vs. implicit). In Table 14 the delta percentage of the optimum solution attained through the several simulation runs of CENEO from the worst and the best solution carried out through ENEO is shown. Therefore, we can appreciate how the solutions of CENEO approximate the optimum achieved through the explicit enumeration strategy.

Table14: relative variation [%] among optimal CENEO solutions and best/worst ENEO solutions.

Sim run	TD [PCU-h/h]	Delta to Best ENEO [%]	Delta to Worst ENEO [%]	CENEO - SA – CT&PDM junction 1 Optimum sequence	junction 2 Optimum sequence	junction 3 Optimum sequence
1	22.06	3	-43	**123**	**1432**	**123**
2	21.41	0	-47	**123**	**1234**	**321**
3	23.32	9	-35	**123**	**1432**	**321**
4	23.07	8	-36	**123**	**1432**	**321**
5	23.33	9	-35	**123**	**1432**	**123**
6	21.58	1	-46	**123**	**1234**	**321**
7	22.47	5	-40	**321**	**1234**	**321**
8	21.63	1	-45	**123**	**1234**	**321**

Summing up, considering that each simulation run in ENEO takes about 2 minutes whilst each simulation run in CENEO takes about 18 minutes, it can be observed (considering that only 16 feasible combination of sequences exist) that: 1) ENEO needs 32 minutes plus 16 minutes to get the approximate optimum and refine it with an improvement of 6% (which is affected by the stochastic nature of the adopted algorithm); 2) against, CENEO needs 18 minutes (and so saving 30 minutes with respect to ENEO) to get the approximate optimum, yielding poorer solution with respect to the best of ENEO even though with a discrepancy which is always less than 10%, and better solution with respect to the worst of ENEO, with a discrepancy which is always more than at least 35%.

It should be stressed that such a comparison results pointless when larger scale network are considered and an higher number of feasible sequences are identified (see case study B).

2.2 CASE STUDY (B)

In this section the application of the scheduled synchronisation problem to a more complex test network is shown. The network layout is shown in Figure 6, while the stage compositions and the equivalence classes of sequences are summarised in Figure 7.

Entry-exit flows, turning percentages of sink links and traffic flow paths are shown in Table15, Table16 and Table17 respectively.

As in the previous section, let m be the number of junctions in the network equal to 5 and n_i the number of feasible stages for each junction i (equal to 2 for junction 1, 3 for junction 2 and 4 for junction 3,4,5), and let us assume that all stages are compulsory for simplicity's sake, there are $(n_i - 1)!$ classes of equivalent sequences for each junction i (i.e. 1 class for junction 1, 2 classes for junction 2 and 6 classes for junction3,4,5), thus, there are $\prod_{i=1,\, m} (n_i - 1)!$ combinations of classes, equal to 432, to be analysed.

Unlike case study A, no comparison between explicit (ENEO) and implicit (CENEO) enumeration approaches are carried out and only the second one is adopted. That is because in doing so (to match the results attained by CENEO with those of ENEO) we need to explore 432 combinations of sequences, all of which require us to address a synchronisation, thus, a complete simulated annealing implementation. In this way, via the explicit enumeration we would achieve 24x432= 10368 traffic flow simulations and so a number of solutions which is too great to explore, which leads to a unpractical pursuable approach.

- place Fig.6 # about here

- place Fig.7 # about here

Table 15: entry-exit flows

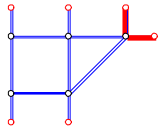
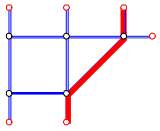
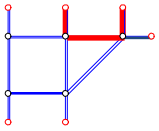
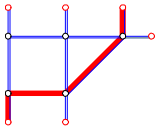
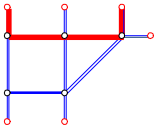
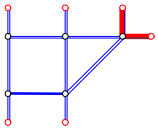
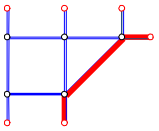
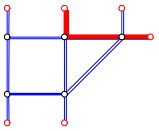
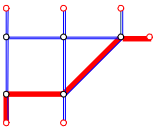
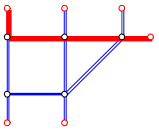
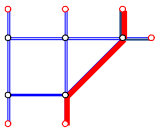
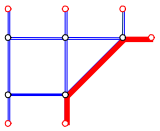
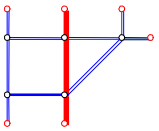
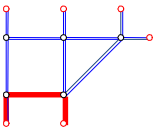
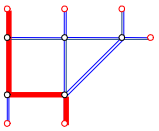
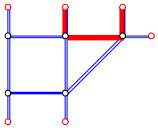
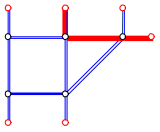
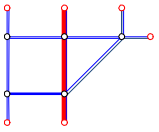
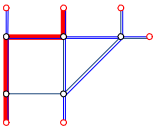
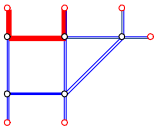
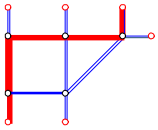
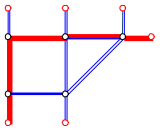
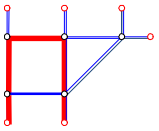
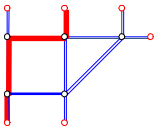
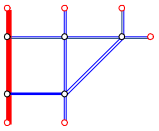
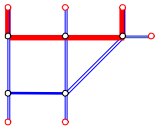
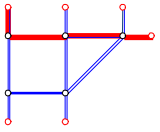
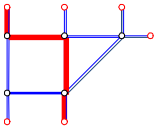
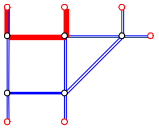
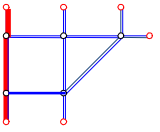
entry/exit	x1	x2	x3	x4	x5	x6	Total
x1	0	70	80	75	78	82	385
x2	45	0	67	63	67	74	316
x3	55	68	0	67	72	56	318
x4	48	75	77	0	74	39	313
x5	71	78	34	96	0	81	360
x6	74	46	88	37	93	0	338
Total	293	337	346	338	384	332	2030

Table 16: turning percentages

Source	turn [%]		
link	Ahead	Left	Right
x1-5	47	39	14
x2-5	43	42	15
x4-4	39	25	36
x3-3	21	40	39
x6-2	28	72	-
x5-1	100	-	-

- *exclusive turn lane

Table 17: traffic flow patterns

to / from	x1	x2	x3	x4	x5	x6
x1						
x2						
x3						
x4						
x5						
x6						

-

In order to analyse the effect of the neighbourhood generation a comparative analysis was worked out to handle the number of internal algorithm iterations (and maintaining the number of external iterations to 24) in such a way to investigate only the neighbourhood exploitation (considering that its size increases congruently with *ii*) which affects the significance of the error percentage obtained within each external iteration. Therefore, (as shown in Figure 8, Figure 9 and Figure 10) the trends of the performance indicators (in terms of network TD) achieved considering different numbers of internal iterations were evaluated (in this case *ii*=3; 5 and 8) in order to verify the solution range of variation. To this aim, a box plot is represented in which internal (int), maximum (max) and minimum (min) performance indicator, as well as current optimum (c.o) are shown.

The second analysis aims to equalise the time consuming independently from the neighbourhood generation, thus, the number of external iterations was constrained in order to fix an upper bound for the number of algorithm implementations (as the product of the internal and external iterations). In particular, it was assumed that such a bound was nearly a third of the total number of feasible combinations. On that basis, as shown in Figure 11 we maintain 24 external iterations when *ii*=5 whilst we reduce *ei* to 15 when *ii*=8 and we extend *ei* to 40 when *ii*=3 .Such an assumption implies that different temperature decrements have to be considered.

- place Fig. 8# about here

- place Fig.9 # about here

- place Fig.10 # about here

The results obtained for the different classes of internal iterations were compared considering three measures of performance (see Table 18)

1) the optimum carried out in terms of TD;

2) the running time of the algorithm;

3) the mean range of variation of the solutions carried out during the internal iterations.

Table 18: performance indicators w.r.t. different values of internal iterations (ii=3, 5 and 8)

internal iterations	running time [min]	TD [PCU h/h]	mean internal range [%]
3	12	25.03	16.46
5	21	19.76	32.01
8	30	19.87	35.11

It should be observed that the running time grows linearly with respect to the number of internal iterations. Both the optimum achieved by the algorithm (in terms of TD) and the mean internal range of variation (higher values result in a larger portion of the search space being explored) are noticeably better when *ii*=5 and *ii*=8 are considered as input. Such a result is due to the fact that presumably only in few cases a significant effect could be observed by switching the combination of stage sequences and then no large internal refining are necessary. On that basis, in this case study the choice of considering the input parameter as the class of 5 internal iterations results as the best one, allowing a considerably safe in computation time and ensuring contemporary promising optimal solutions.

In order to verify the stability of the solution achieved by CENEO, as in case study A, a measure of the internal dispersion has been provided. In particular, for a fixed number of internal iterations equal to 5, several runs[7] have been carried out, evaluating the delta percentage to best solution. Results shown in Table 19 and Table 20 make clear that no meaningful variation among solutions arises even though the stochastic nature of the optimisation algorithm. As a matter of fact, as in case study A (Table 7 and Table 8), the fluctuations are less than the 10% of the best solution.

In order to equalise the time consuming without assuming any variation of the number of internal iterations we perform, as previously described, a further analysis (see Figure 11) which consists in adopting three different temperature decrements, each one derived from

[7]Through a preliminary analysis it was observed that differently from case study A a higher number of iterations was required in the purpose of reaching the stability of the solutions.

a different temperature range. In particular, we assume that the start and the final temperature are fixed but we vary the number of temperature states in order to generate different decrement lengths. The case of 5 internal iterations was considered as reference, thus, we obtain 15 external iterations when *ii*=8 and 40 external iterations when *ii*=3.

- place Fig.11 # about here

As expected when *ii*=8 and *ei*=15, as high temperature decrements are adopted only a few search spaces are explored, thus a proxy of a local search is performed yielding a poor optimum result. However, when *ii*=3 and *ei*=40 low temperature decrements are applied in order to randomise the search for a number of external iterations until the temperature was cool enough to make the algorithm act as a simulated annealing; as a matter of fact, the starting point of the cooling process is not good enough to generate a solution which is better with respect to the reference one attained considering *ii*=5 and then *ei*=24. Nevertheless, even though it is quite clear that a large number of internal iterations, on the basis of the trade-off between the results obtained in the two aforementioned analyses, leads to worse global performances (especially in terms of time consuming), no general considerations can be made when lowering the number of external iterations since the results are affected by a certain randomness.

Table 19: optimal sequences computed through CENEO over several runs)

CENEO							
junction 1	junction 2	junction 3	junction 4	junction 5	TD [PCU-h/h]	Sim run	Delta to best [%]
Opt. seq.	Opt. seq.	Opt. seq.	Opt. seq.	Opt. seq.			Internal dispersion
12	123	1342	1423	1234	20.98	1	+5.82
12	123	1324	1342	1432	20.92	2	+5.54
12	123	1234	1432	1324	20.05	3	+1.45
12	123	1342	1423	1234	20.46	4	+3.42
12	123	1234	1432	1324	20.77	5	+4.86
12	123	1324	1342	1432	21.86	6	+9.61
12	132	1234	1234	1342	20.92	7	+5.54
12	123	1234	1432	1324	19.76	8	0.00
12	132	1234	1234	1342	20.66	9	+4.36
12	123	1324	1342	1432	20.82	10	+5.09
12	123	1342	1423	1234	21.18	11	+6.70
12	123	1234	1432	1324	20.84	12	+5.18
12	123	1234	1432	1324	20.39	13	+3.09
12	132	1234	1234	1342	21.26	14	+7.06
12	123	1324	1342	1432	21.54	15	+8.26
12	123	1234	1432	1324	19.76	16	+0.00

12	123	1342	1423	1234	21.84	17	+9.52
12	123	1324	1342	1432	21.47	18	+7.96
12	123	1234	1432	1324	20.98	19	+5.82
12	132	1234	1234	1342	21.33	20	+7.36

Table 20: optimal greens and offsets computed through CENEO over several runs)

CENEO																							
junction 1		junction 2				junction 3					junction 4					junction 5					TD [PCU-h/h]	Sim run	Delta to best [%] Internal dispersion
t_1	t_2	t_1	t_2	t_3	ϕ_2	t_1	t_2	t_3	t_4	ϕ_3	t_1	t_2	t_3	t_4	ϕ_4	t_1	t_2	t_3	t_4	ϕ_5			
66	34	45	38	17	55	27	17	30	26	80	23	26	30	21	96	24	29	20	27	59	20.98	1	+5.82
65	35	43	41	16	59	26	14	32	28	81	24	29	27	20	90	22	26	21	31	61	20.92	2	+5.54
65	35	42	39	19	50	28	16	30	26	65	26	25	28	21	71	23	26	22	29	38	20.05	3	+1.45
63	37	42	39	19	54	25	19	32	24	69	22	30	25	23	82	20	24	23	33	55	20.46	4	+3.42
64	36	43	41	16	54	25	15	32	28	83	22	29	30	19	95	23	24	22	31	63	20.77	5	+4.86
63	37	45	35	20	63	27	16	28	29	94	25	29	24	22	21	23	27	23	27	74	21.86	6	+9.61
65	35	43	41	16	59	26	14	32	28	81	24	29	27	20	90	22	26	21	31	61	20.92	7	+5.54
66	34	42	40	18	50	27	14	30	29	64	27	24	26	23	67	24	26	21	29	32	19.76	8	0.00
62	38	43	38	19	54	23	18	33	26	73	22	30	28	20	88	23	24	23	30	66	20.66	9	+4.36
65	35	42	39	19	57	25	15	32	28	81	24	30	26	20	92	22	26	23	29	61	20.82	10	+5.09
65	35	40	39	21	53	25	15	31	29	80	26	30	24	20	97	23	26	23	28	69	21.18	11	+6.70
63	37	41	40	19	56	26	15	33	26	76	23	27	27	23	96	22	26	24	28	72	20.84	12	+5.18
63	37	41	39	20	55	25	16	35	24	71	23	31	25	21	82	23	23	22	32	56	20.39	13	+3.09
64	36	42	39	19	57	25	14	30	31	81	24	27	26	23	94	25	25	22	28	59	21.26	14	+7.06
64	36	44	38	18	51	28	15	31	26	80	27	31	24	18	11	21	26	24	29	77	21.54	15	+8.26
66	34	42	40	18	50	27	14	30	29	64	27	24	26	23	67	24	26	21	29	32	19.76	16	+0.00
63	37	43	37	20	59	27	19	28	26	93	25	30	23	22	21	24	27	22	27	81	21.84	17	+9.52
65	35	45	38	17	56	28	15	29	28	81	25	32	25	18	8	21	26	23	30	70	21.47	18	+7.96
66	34	45	38	17	55	27	17	30	26	80	23	26	30	21	96	24	29	20	27	59	20.98	19	+5.82
64	36	42	37	21	53	25	14	31	30	82	24	28	24	24	100	25	24	23	28	64	21.33	20	+7.36

3 CONCLUSIONS

This paper focuses on a general approach to Network Signal Setting Design (NSSD) including green scheduling, green timing and coordination into the so-called scheduled synchronisation. The proposed method, called CENEO (ComplEte NEtwork Optimisation), is an extension of the synchronisation method and the traffic flow model proposed in Cantarella et al. (2015) that requires the stage sequence as input data and optimises stage lengths and offsets only. CENEO, to authors' knowledge, is the first method available in literature for scheduled synchronisation.

In order to discuss the efficiency of CENEO and to calibrate the parameters of the SA algorithm a three and a five nodes networks have been analysed. The iterations were distinguished in external and internal; the former representing the number of temperature states adopted to apply the simulated annealing scheme, the latter representing the number of combinations (randomly selected from the equivalence classes) tested within each external iteration together with the current greens and offsets. In particular, firstly, for a fixed value of external iterations, the effect of different values of internal iterations (3, 5 and 8) were tested (see Figure 8, Figure 9 and Figure 10); then the external and internal iterations pairs were obtained by fixing their product. Results of both analyses indicate that higher values of internal iterations not necessarily improve general performance.

In order to discuss the effectiveness of CENEO the results performed by CENEO have been also compared with those carried out by explicitly enumerating all possible stage sequences and then analysing them for the three nodes network only (explicit enumeration is, in fact, suitable only for a very small simple network as noted in section 3). The values of the performance indicators provided by CENEO over 8 simulations are in one case the best feasible solution, in four cases within an error less than 5% and in three cases within an error less than 10% (see Table 14) of the optimal solution generated through complete enumeration of all combinations.

In conclusion, CENEO is an efficient and effective method for scheduled synchronisation, it is presumably suitable for large scale application, and implementation in commercial software suites. Moreover, it can also be generalised to Signal Setting Global Optimisation with equilibrium constraints as well as for Urban Network Design Problem, including lane allocation.

Possible enhancements worthy of research are: testing other traffic flow models (e.g. a mesoscopic traffic flow model; see Di Gangi et al., 2016), the investigation of hybrid meta-heuristics (i.e. population based with neighbourhood based methods) presumably more suitable for large scale networks, NSSD based on reserve capacity maximisation at network level. Extensions of Phase-based and Lane-based methods to NSSD also seem interesting research perspectives.

Acknowledgements

This research has been partially supported by the University of Salerno, under PhD program on transportation (Ph.D. School in Environmental Engineering), local grant n. ORSA132945 - 2013, local grant n. ORSA165221 - 2016, and under 'APPS4SAFETY – PON03PE_00159_3'. We wish to thanks anonymous reviewers who provided very helpful comments to improve the paper. For our friendship policy, the authors' order is organised as in follows: the first author is a PhD Candidate whose research thesis mainly focused on the topic of the paper, the other authors are in alphabetical order.

References

Allsop, R.E., 1971. SIGSET: a computer program for calculating traffic signal settings. Traffic Engineering and Control 13 (2), 58–60.

Allsop, R.E., 1975. Computer program SIGCAP for assessing the traffic capacity of signal-controlled road junctions –description and manual for users, Transportation Operations Research Group Working Paper, Vol. 11, University of Newcastle upon Tyne.

Binning, J. C., Crabtree, M. R., & Burtenshaw, G. L., 2010.Transyt 14 user guide. Transport Road Laboratory Report nr AG48. APPLICATION GUIDE 65 (Issue F).

Bron, C., & Kerbosch, J., 1973. Algorithm 457: finding all cliques of an undirected graph. Communications of the ACM, 16(9), 575-577.

Burrow, I.J., 1987. OSCADY: a computer program to model capacities, queues and delays at isolated traffic signal junctions, TRRL Report, Vol. 105, Transport and Road Research Laboratory, Crowthorne.

Cantarella, G. E., de Luca, S., Di Gangi, M., Di Pace, R., &Memoli, S. (2014). Macroscopic vs. mesoscopic traffic flow models in signal setting design. In Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference 2221-2226

Cantarella, G. E., de Luca, S., Di Pace, R., & Memoli, S. (2015). Network Signal Setting Design: meta-heuristic optimisation methods. Transportation Research Part C: Emerging Technologies, 55, 24-45.

Daganzo, C.F., 1994. The cell-transmission model. Part 2: Network traffic, University of California, Berkeley, California.

Di Gangi, M., Cantarella, G. E., Di Pace, R., & Memoli, S. (2016). Network traffic control based on a mesoscopic dynamic flow model. Transportation Research Part C: Emerging Technologies, 66, 3-26.

Friesz, T.L., Han, K., Neto, P.A., Meimand, A., Yao, T., 2013. Dynamic user equilibrium based on a hydrodynamic model. Transportation Research Part B 47, 102–126.

Gallivan, S. Heydecker, B. 1988 Optimising the control performance of traffic signals at a single junction. Transportation Research Part B: Methodological, 22(5) 357-370.

Gayah, V.V., Daganzo, C.F., 2012. Analytical capacity comparison of one-way and two-way signalized street networks. Transportation Research Record 2301, 76–85

Gu, W., Gayah, V.V., Cassidy, M.J., Saade, N., 2014. On the impacts of bus stops near signalized intersections: models of car and bus delays. Transportation Research Part B 68, 123–140.

Hadi, M. A. & Wallace, C. E., 1993. Hybrid genetic algorithm to optimize signal phasing and timing Transportation Research Record 1421, 94-103.

Hadi, M. A., & Wallace, C. E. 1994. Optimization of signal phasing and timing using cauchy simulated annealing. Transportation Research Record, (1456), 64-71.

Han, K., & Gayah, V. V. 2015. Continuum signalized junction model for dynamic traffic networks: Offset, spillback, and multiple signal phases. Transportation Research Part B: Methodological, 77, 213-239.

Han, K., Gayah, V., Piccoli, B., Friesz, T.L., Yao, T., 2014. On the continuum approximation of the on-and-off signal control on dynamic traffic networks. Transportation Research Part B 61, 73–97

Improta, G., & Cantarella, G. E. 1984. Control system design for an individual signalized junction. Transportation Research Part B: Methodological, 18(2), 147-167.

Lam, W.H.K, Poon, A.C.K. and Mung, G.K.S. (1997) Integrated model for lane-use and signal-phase designs. Journal of Transportation Engineering, ASCE, 123, 114-122.

Liu, R., & Smith, M., 2015. Route choice and traffic signal control: A study of the stability and instability of a new dynamical model of route choice and traffic signal control. Transportation Research Part B: Methodological, 77, 123-145.

Metropolis, N. Rosenbluth, A.W. Rosenbluth, M.N. Teller, A.H. Teller, E., 1953Equation of state calculation by fast computing machines J. Chem. Phys.. 1087–1092.

Park, B., B., Messer, J., C., Urbanik II, T., 2000. Enhanced Genetic Algorithm for Signal-Timing Optimization of Oversaturated Intersections. Transportation Research record 1727, no.00.1661.

Silcock, J.P., 1997. Designing signal-controlled junctions for group-based operation. Transportation Research Part A: Policy and Practice, 31(2): 157-173.

Smith, M. J., Liu, R., & Mounce, R., 2015. Traffic control and route choice: Capacity maximisation and stability. Transportation Research Part B: Methodological, 81, 863-885.

Sun, W., Wu, X., Wang, Y. & Yu, G. (2015), A continuous- flow-intersection-lite design and traffic control for oversaturated bottleneck intersections, Transportation Research Part C: Emerging Technologies, 56, 18–33.

Webster, F.V., 1958. Traffic signal settings, Road Research Technical Paper, No. 39, HMSO, London.

Wong, C.K. and Wong, S.C., (2003). Lane-based optimization of signal timings for isolated junctions. Transportation Research Part B, 37, 291-312.

Wong, C.K., Wong, S.C. and Tong, C.O., (2000). Lane-based optimization method for maximizing reserve capacity of isolated signal-controlled junctions. Proceeding of the Fifth Conference of Hong Kong Society for Transportation Studies, 2 December, Hong Kong, pp. 176-184.

Wong, S.C., 1996. Group-based optimisation of signal timings using the TRANSYT traffic model. Transportation Research Part B: Methodological, 30(3): 217-244.

Wong, S.C., 1997. Group-based optimisation of signal timings using parallel computing. Transportation Research Part C: Emerging Technologies, 5(2): 123-139.

Wong, S.C., Wong, W.T., Leung, C.M., Tong, C.O., 2002. Group-based optimization of a time-dependent TRANSYT traffic model for area traffic control. Transportation Research Part B: Methodological, 36(4): 291-312.

Yu, C., Ma, W., Lo, H. K., & Yang, X. (2016). Robust Optimal Lane Allocation for Isolated Intersections. Computer Aided Civil and Infrastructure Engineering. DOI: 10.1111/mice.12236

Zhou, Y. & Zhuang, H. (2014), The optimization of lane assignment and signal timing at the tandem intersection with pre-signal, Journal of Advanced Transportation, 48(4), 362– 76.

LIST OF NOTATIONS

We list here the parameters used in the paper

n	number of candidate stages
n_o	number of optional candidate stages
n_c	number of compulsory candidate stages
m	number of junctions in the network
c	cycle length (s)
t_j	length of stage j (s)
t_{ar}	all red period (s)
l_k	lost time for approach k (s)
g_k	effective green for approach k (s)
q_k^j	arrival flow for approach k (pcu/hr) of the generic junction j
s_k^j	saturation flow for approach k (pcu/hr) of the generic junction j
ϕ_{ih}	link offset between any two adjacent junctions i and h (s)
ϕ_i	node offset at junction i (s)
TD_k^j	single junction total delay (pcu/hr x hr) on approach k of junction j
$Ciq_k^j(t)$	cumulated input flows (pcu/hr) through the stop line of approach k of junction j, in the subsequent sub-intervals t
$Coq_k^j(t)$	cumulated output flows (pcu/hr) through the stop line of approach k of junction j, in the subsequent sub-intervals t
$\Delta\tau$	the sub-interval of the time of analysis T_a
DTD_k^j	Deterministic Total Delay (pcu/hr x hr) cumulated in the interval $[0,T_a]$ for approach k of junction j
$SOTD_k^j$	Stochastic Oversaturation component of Total Delay (pcu/hr x hr) cumulated in the interval $[0,T_a]$ for approach k of junction j
TD	network total delay (pcu/hr x hr)
P(c)	probability of acceptance
r	real number randomly drawn from a uniform distribution over [0, 1]
T	current temperature of the system
T_a	time of analysis
T_{init}	initial temperature

T_{fin}	final temperature
T_{it}	temperature at iteration *it*
α	constant (equal to 0.95)
x	vector of greens and offsets
N	number of combination of sequences within the inner loop
s_0	combination of sequences at iteration 0
$x_0 = f(g_0,\varphi_0)$	green and offsets generation at iteration 0
s_{it}	combination of sequences at iteration *it*
$x_{it} = f(g_{it},\varphi_{it})$	green and offsets generation at iteration *it*
$c.o = f(x_{it},s_{it})$	current optimum solution at iteration *it*
$c.n = f(x_{it+1},s_{it+1})$	current new solution at iteration *it*
$\Delta c = c.n - c.o$	delta between current new and current optimum solutions
b	binary variable (that may get one of two values)
t	ternary variable (that may get one of three values)