Appendix F. Optimization of Offsets and Cycle Length Using High Resolution Signal Event Data

Attached document:

C.M. Day and D.M. Bullock. "Optimization of Offsets and Cycle Length Using High Resolution Signal Event Data." Working paper, SPR-3409, Joint Transportation Research Program, August 2011.

Optimization of Offsets and Cycle Length Using High Resolution Signal Event Data

Christopher M. Day
Purdue University

Darcy M. Bullock
Purdue University

Working Paper
August 31, 2011

INTRODUCTION

This document explains the prototype model for cycle length and offset optimization developed for SPR 3409. This model, referred to in this document as "Traffic Signal Model 3409" (TSM 3409), was coded in Processing (1), a Java-based, object-oriented development environment intended for rapid, overhead-free prototyping, with emphasis on graphical output. The model has the following functions:

1. To fully describe and estimate the state of a traffic network under a given combination of cycle, offsets, splits, and phase sequence, with a dual-ring eight-phase traffic signals, for a network containing an arbitrary number of traffic signals using a macroscopic traffic model.

2. To optimize the cycle length and offsets of an arterial signal system.

The purpose of developing this model is to demonstrate how real data can be integrated into the optimization process, rather than to propose a new traffic model. The specific application investigated in this paper is arterial signal systems with channelized turns under phase sequences that conform to allowable configurations in the dual-ring eight-phase scheme. This conforms to the emphasis of the research in SPR 3409. However, it would not be difficult to modify the model to handle half/double cycles, shared lanes, non-arterial systems, and other such scenarios. The traffic model is based on TRANSYT (2).

TRAFFIC MODEL

Basic Elements

The traffic system is considered as a series of road segments that are joined at nodes. In TSM 3409, all nodes are modeled as signalized intersections. Each segment may accommodate as many as three inflows and three outflows; each departing path is called a "link". Figure 1 illustrates inflows and outflows on a segment. The arrival profiles for each "link" that is a member of the segment have the same normalized shape, but are normalized to the expected arrival volume (as determined from field counts).

The links are the fundamental elements for estimating the network state. The network state is described by the set of link cyclic flow profiles that describe the expected arriving and departing traffic flows that occur in the system. These profiles are calculated using the Robertson platoon dispersion model as implemented in TRANSYT (2).

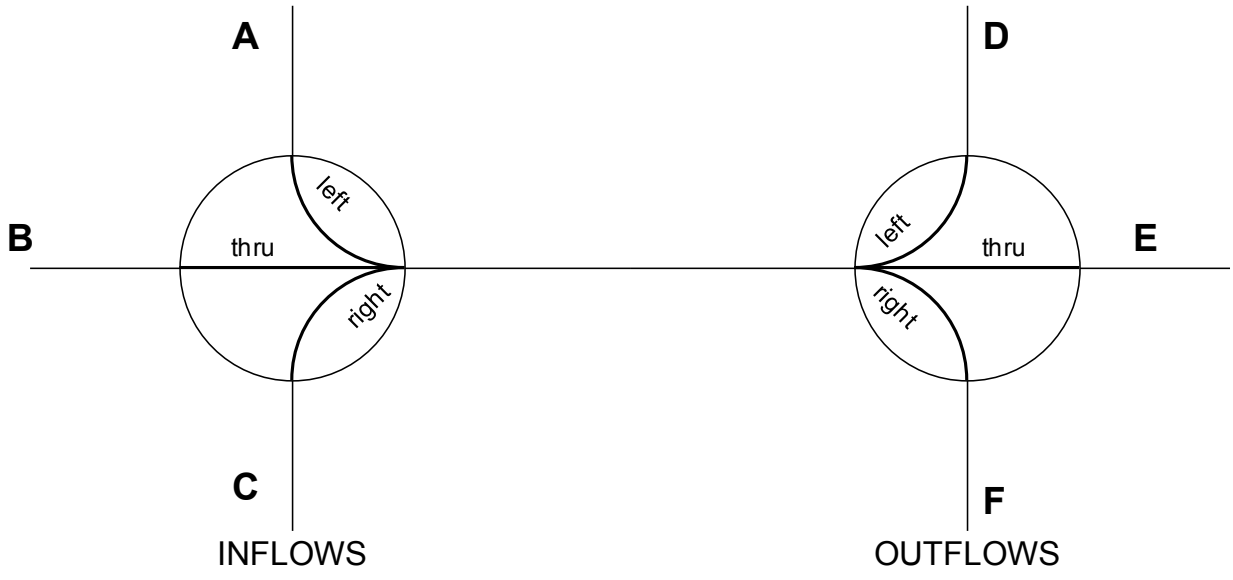


Figure 1: Segment flows.

Network State Variables and Measures of Effectiveness

The macroscopic traffic state of each link in the system is contained in the variables shown in Table 1. Using Robertson's methodology proposed in the original formulation of TRANSYT (2), the *arrivals* on each link are modeled as the sum of *departures* from the upstream links; the departure profiles are calculated by assuming uniform arrivals on each upstream turn. This eliminates the need for an iterative calculation process that would be required if "actual" departure profiles (that is, estimated from calculating departures of the upstream link using *its* upstream departures—and so on, until reaching the edges of the system where arrivals are uniform). This is equivalent to the methodology used in TRANSYT.

During SPR-3409, models based on an iterative and non-iterative macroscopic models were tested during the coding process; the difference in models was a slight impact on the shape of the tails of platoons in the vehicle distributions. This paper presents only the results from non-iterative model, as the iterative model imposes limitations on the types of optimization strategies that can be used (because it breaks the assumption of nearest-neighbor interactions). It is worthwhile to reflect momentarily that our primary objective in the context of SPR 3409 is to identify the optimal cycle length in the system.

Table 1. List of variables used in TSM 3409.

Variable	Meaning
i	Bin number
j	ID number of upstream link (j = 1 … 3)
k	Ring position
$R^*[X, i]$	Expected phase in ring X during i^{th} bin (arbitrary timeline)
$R^0[X, i]$	Expected phase in ring X during i^{th} bin (local clock)
$R[X, i]$	Expected phase in ring X during i^{th} bin (system clock)
SEQ[X, k]	ID of phase in ring X, position k
χ_X	Coordination reference phase, ring X
C	Cycle Length
θ	Offset
SPT[φ]	Split for phase φ, s
Y[φ]	Clearance time for phase φ (yellow and all-red), s
$g[i]$	Status of green during i^{th} bin
s	Saturation flow rate (veh/s)
$c[i]$	Capacity, i^{th} bin
$U_j[i]$	Upstream depatures, i^{th} bin, link j (Departures at link j based on uniform arrivals)
$U_0[i]$	Departures from the current link of interest (i^{th} bin), based on uniform arrivals
$D[i]$	"Real" departures, i^{th} bin (calculable, but not used in the model presented here)
$q[i]$	Queue length (based on uniform arrivals)
$A_U[i]$	Uniform arrivals, i^{th} bin
T	Link travel time (number of bins)
F	Robertson dispersion parameter, F
$a_0[i]$	"Real" arrivals – prior to normalizing for volume
$A_0[i]$	"Real" arrivals – before adjusting for travel time, i^{th} bin
$A_T[i]$	"Real" arrivals – before adjusting for dispersion, i^{th} bin
$A[i]$	"Real" arrivals – after adjusting for dispersion, i^{th} bin
$Q[i]$	Queue length (based on "real" arrivals), i^{th} bin
N_g	Arrivals on green
S	Number of stops
d	Delay
v	Total arrival volume
x	Volume-to-capacity ratio
PI	Performance Index
w	Delay equivalent of each stop (sec)

The calculations of the network state are summarized as follows.

Step 1. First, the ring timers (R^*, R^0, R) are populated. These variables store the phase number of whichever phase is expected to be green at that particular time in cycle. The timing is established such that the first beginning of coordinated green (phase 2 or phase 6) is the local zero ("TS2 first green"). That is, for a given intersection, either phase 2 or phase 6 beginning of green (depending on whichever is earliest) will have its beginning of green in the first bin, and the other phases are adjusted accordingly. Finally, we shift the distribution using the offset in order to get the system time in which the phase is green.

The status of R^* is defined by:

$$R^*[X,i] = \begin{cases} SEQ[X,k'], & \text{if } \sum_{k=1}^{k'} SPT[k] \geq i > \left(\sum_{k=1}^{k'} SPT[k] + SPT[k']\right) \\ 0, & \text{if the above is true AND ring is in clearance : } \\ & i + \sum_{k=1}^{k'} SPT[k] > \left(SPT[k'] - Y[k']\right) \\ 0, & \text{if } i > C \end{cases} \qquad \text{Equation 1}$$

In the above, k' represents whichever phase is currently in use at bin i for ring X.

The local cycle state (R^0) is obtained from:

$$R^0[X,i] = R^*[X,(i-z) \bmod C] \qquad \text{Equation 2}$$

$$z = \min\left\{ \min_i \{R^*[1,i] = \chi_1\}, \quad \min_i \{R^*[2,i] = \chi_2\} \right\} \qquad \text{Equation 3}$$

This simply means we find the earliest occurrence of χ_1 or χ_2, and shift the entire ring state accordingly.

The final step is to adjust the ring state one more time to account for the offset, converting the timeline from the local clock to the system clock.

$$R[X,i] = R^0[X,(i-\theta) \bmod C] \qquad \text{Equation 4}$$

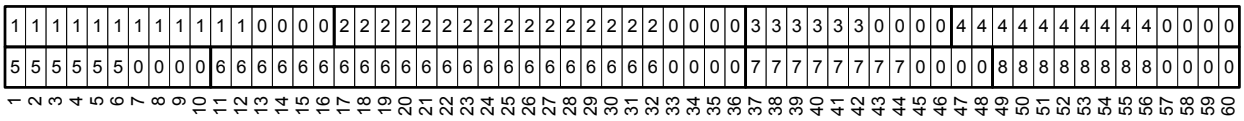
Figure 2 provides a graphical example of how the ring state calculation is worked out in TSM 3409, for a cycle length of 60 seconds. The timing parameters (Figure 3a) are taken directly from the programmed signal timing plan (or desired parameters). An "arbitrary timeline" ring state is then drawn from this information (Figure 3b) using the rules in Equation 1. In this example (and typical of most arterial systems), the coordination reference phases are $\chi_1 = 2$ and $\chi_1 = 6$; the earlier first occurrence ("TS2 first green") is for phase 6, occurring in bin 11. The entire timeline is shifted so that this is redefined as bin 1 (Figure 3c); this represents what the controller will actually use as "local zero" during coordinated operations. Finally, the entire timeline is shifted according to the intersection offset (Figure 3d), to make it relevant to the rest of the coordinated system.

This paper does not explore the use of double or half cycling in the investigation of optimization methodologies. However, it would be quite simple to model double/half cycles by taking the results of Equation 1 for $C_H = C/2$ and applying the following formula to $(C_H + 1) \geq i > 2C_H$.

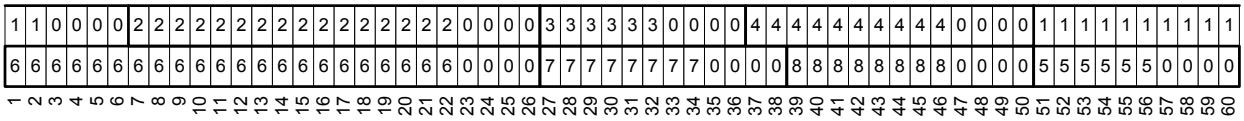
$$R^*[X,i] = R[X,i - C_H] \qquad \text{Equation 5}$$

SEQ[1,1] = 1 SPT[1]=16,Y[1]=4	SEQ[1,2] = 2 SPT[2]=20,Y[2]=4	SEQ[1,3] = 3 SPT[3]=10,Y[3]=4	SEQ[1,4] = 4 SPT[4]=14,Y[4]=4
SEQ[1,2] = 5 SPT[5]=10,Y[5]=4	SEQ[2,2] = 6 SPT[6]=26,Y[1]=4	SEQ[2,3] = 7 SPT[7]=12,Y[7]=4	SEQ[2,4] = 8 SPT[8]=12,Y[8]=4

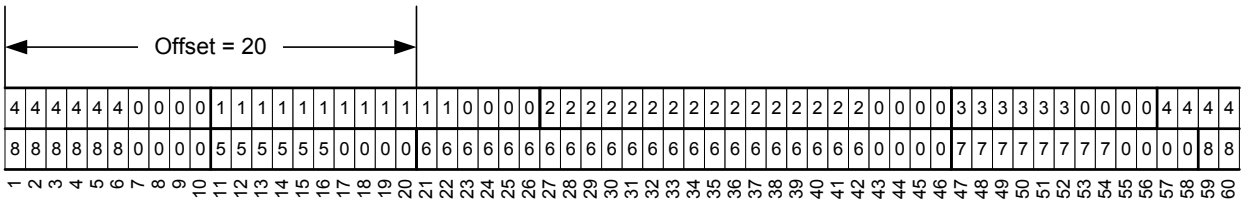
(a) Phase sequence and timing parameters.



(b) Arbitrary timeline ring state (R^*).



(c) Adjusted to local cycle using TS2 first green (R^0) .



(d) Adjusted to global cycle (R).

Figure 2: Obtaining global phase times at each intersection.

Step 2. Next, the actual green status of each individual movement is found by looking at the ring timers and storing a value of "1" (green) whenever the controlling phase (φ') is contained in the ring timer variable.

$$g_i = \begin{cases} 1, & R[X,i] = \varphi' \\ 0, & \text{otherwise} \end{cases} \qquad \text{Equation 6}$$

This is multiplied by the saturation flow rate to obtain the capacity:

$$c_i = s g_i \qquad \text{Equation 7}$$

Step 3. The cyclic flow profiles throughout the whole network in a two-stage process:

a. In the first stage, we calculate the departure profile at each link based on the assumption of uniform arrivals. In TSM 3409 we call these the "external link states" because the departure profiles are used for other links in the second stage.
b. In the second stage, for all links in the network where non-random arrivals are expected to occur, the arrival profiles are recalculated as the summation of upstream link departures calculated in the first stage. These are called the "internal link states" because the information is not used for other links.

Alternatively, an <u>iterative</u> process directly uses link outflows as the inflows for other links. In this case, a change that occurs at one point in the system will cascade through all subsequent links. The network state is iteratively calculated until there is no additional change in any of the flow profiles; the stopping condition is $Z_k - Z_{k-1} = 0$, where:

$$Z_k = \sum_L \sum_i (i) * D[i]$$

and k is the iteration number and the outer summation is over all links L in the network. The number of iterations (maximum k) required is approximately 15. The remainder of this paper focuses on the <u>non-iterative</u> process.

The cyclic flow profiles for each turn are calculated by the following formulas. The following formulation is essentially the same as Robertson's macroscopic traffic simulation in TRANSYT (2), except where noted.

For the "external" link states (uniform arrival assumption), the arrival-queue-departure process is calculated from:

$$A_U[i] = \frac{v}{C} \qquad \text{Equation 8}$$

$$q[i] = \max\{0, \quad q[i-1] - c[i] + A_U[i]\} \qquad \text{Equation 9}$$

$$U_0[i] = \max\{0, \quad q[i-1] - q[i] + A_U[i]\} \qquad \text{Equation 10}$$

For the "internal" link states (non-uniform arrivals), the arrival-queue-departure process is calculated from:

$$a_0[i] = \sum_{j=1}^{3} U_j[i] \qquad \text{Equation 11}$$

$$A_0[i] = \frac{a_0[i]}{\sum_i a_0[i]} v \qquad \text{Equation 12}$$

The normalization of the arrival profile a_0 to the expected volumes v is not explicitly stated by Robertson (2), but is necessary to get realistic delay estimates.

When no upstream links are defined, then $a_0 = A_0 = A_U$.

The two following equations represent travel along the link, and platoon dispersion:

$$A_T[i] = A_0[(i-T) \bmod C] \quad \text{Equation 13}$$

$$A[i] = F \cdot A_T[i] + (1-F) \cdot A_T[i-1] \quad \text{Equation 14}$$

The queues and departures are calculated in the same way as Equation 9 and Equation 10. In TSM 3409, the output of Equation 16 is not directly used.

$$Q[i] = \max\{0, \quad Q[i-1] - c[i] + A[i]\} \quad \text{Equation 15}$$

$$D[i] = \max\{0, \quad Q[i-1] - Q[i] + A[i]\} \quad \text{Equation 16}$$

Step 4. Finally, the link performance measures are calculated:

$$x = \frac{v}{\sum_i c_i} \quad \text{Equation 17}$$

$$N_g = \sum_i g[i] \cdot A[i] \quad \text{Equation 18}$$

$$S = \sum_i A[i] \cdot \delta_i \quad \text{Equation 19}$$

$$\delta_i = \begin{cases} 1, & (Q[i] > 0) \text{ or } (g[i] = 0) \\ 0, & \text{otherwise} \end{cases} \quad \text{Equation 20}$$

$$d = \sum_i Q[i] \quad \text{Equation 21}$$

$$PI = d + wS \quad \text{Equation 22}$$

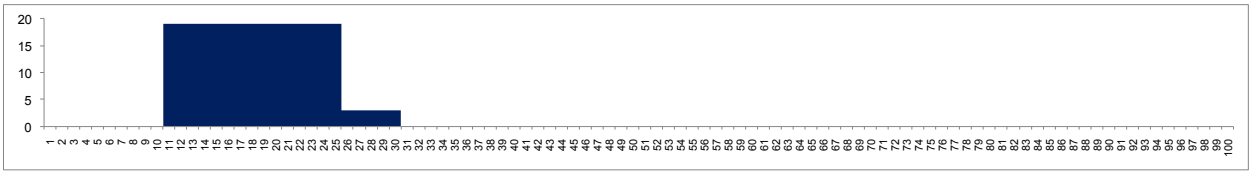
Figure 3 and Figure 4 illustrate the flow profile model graphically.

- Figure 3a, Figure 3b, and Figure 3c, show three upstream departure flows for a particular link, which are summed together to obtain the total combined arrival flow (a_0) shown in Figure 3d. This flow is then normalized, translated, and dispersed using Equation 12, Equation 13, and Equation 14, to produce the arrival flow (A_0) in Figure 3e.

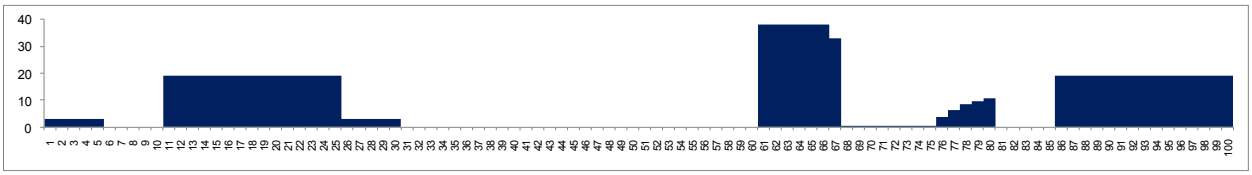
- The status of green is obtained from the ring state using Equation 6 (Figure 4a). In this example, the controlling phase is coordinated phase 2, and an offset of 45 seconds is in use. This is converted to capacity using Equation 7 (Figure 4b), and combined with the arrival profile (Figure 4c) to estimate the queue using Equation 15 (Figure 4d).

(a) Entries from upstream through movement (U_1).

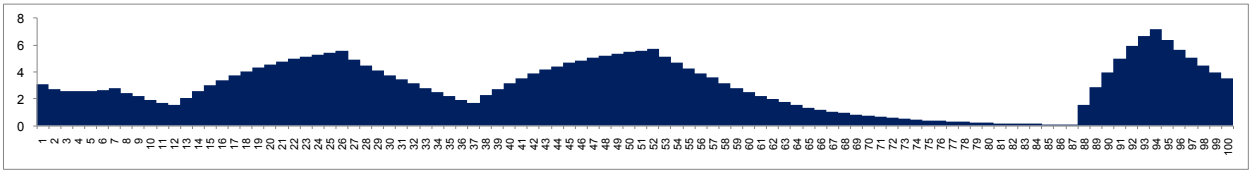
(b) Entries from upstream right turn (U_2).



(c) Entries from upstream left turn (U_3).

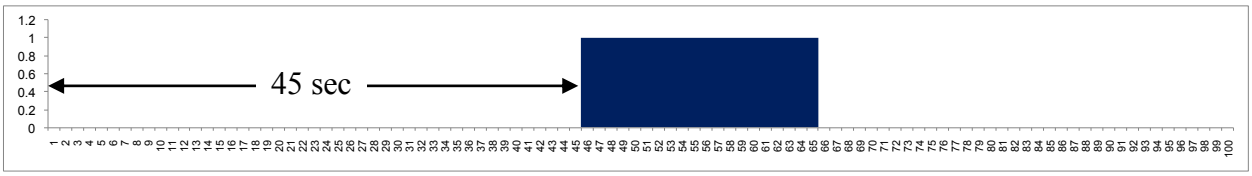


(d) Combined upstream arrivals (a_0).

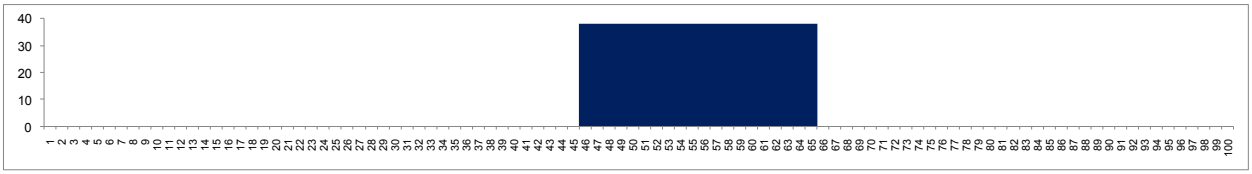


(e) Normalized, translated, and dispersed arrival flow (A),
($T = 27$ and $F = 0.113$).

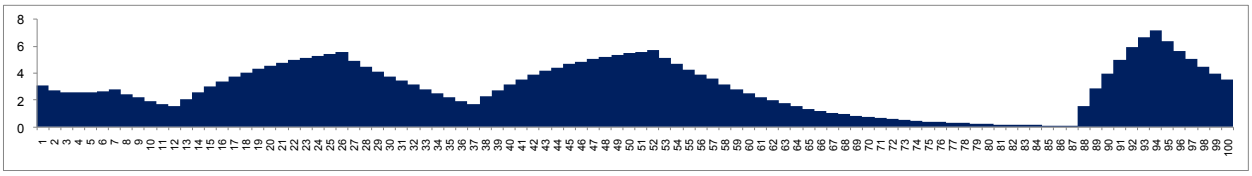
Figure 3: Graphical example of flow profile concept: calculating arrivals.



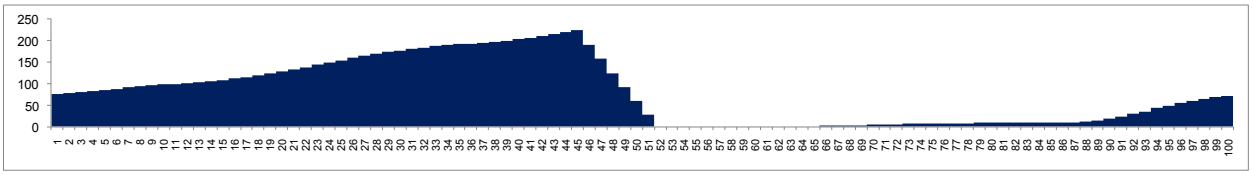
(a) Green status (*g*).



(b) Capacity (*c*).



(c) Arrival profile (*A*).



(d) Queue profile (*Q*).

Figure 4: Graphical example of flow profile concept: queue model.

MODEL IMPLEMENTATION

Model Implementation and Performance

The model was implemented as a program (TSM 3409) which was coded using Processing (1). Although the model in its basic implementation in the program is generally unlimited in terms of system geometry, the output display in this particular application was constrained to arterial systems, and the scope of optimization was limited to the arterial through movements. Figure 5 is a screenshot of the program during runtime for a nine-intersection system. The upper left hand corner of the screen contains a display of the coordinated through movement profiles (detail in Figure 6); the numbers in each box represent delay in seconds and the number of stops. The lower right hand corner of the screen shows a time-space diagram (detail in Figure 7). This is not used for optimization, but facilitates visualization of the current plan. The upper right hand portion of the screen displays the current signal timing plan used in the current model scenario.

The speed of execution for the example system is relatively fast when the non-iterative model is used; To execute a Monte Carlo simulation[1], 156,629 random scenarios were generated in 2h 47m, equating to about 0.0639 sec/scenario. However, this speed is still not terribly efficient for optimization processes that require a large number of scenario estimations. To economize on the computational burden, a full model refresh is executed only when the entire network requires a state update. For some of the algorithms discussed later, local optimization operations could be limited to a sub-network by transferring the relevant information to an abstraction layer; this allowing those algorithms to run much faster by reducing the number of variables to be updated during each step. This is discussed further later.

[1] A battery of random-parameter trials that characterizes the parameter space.

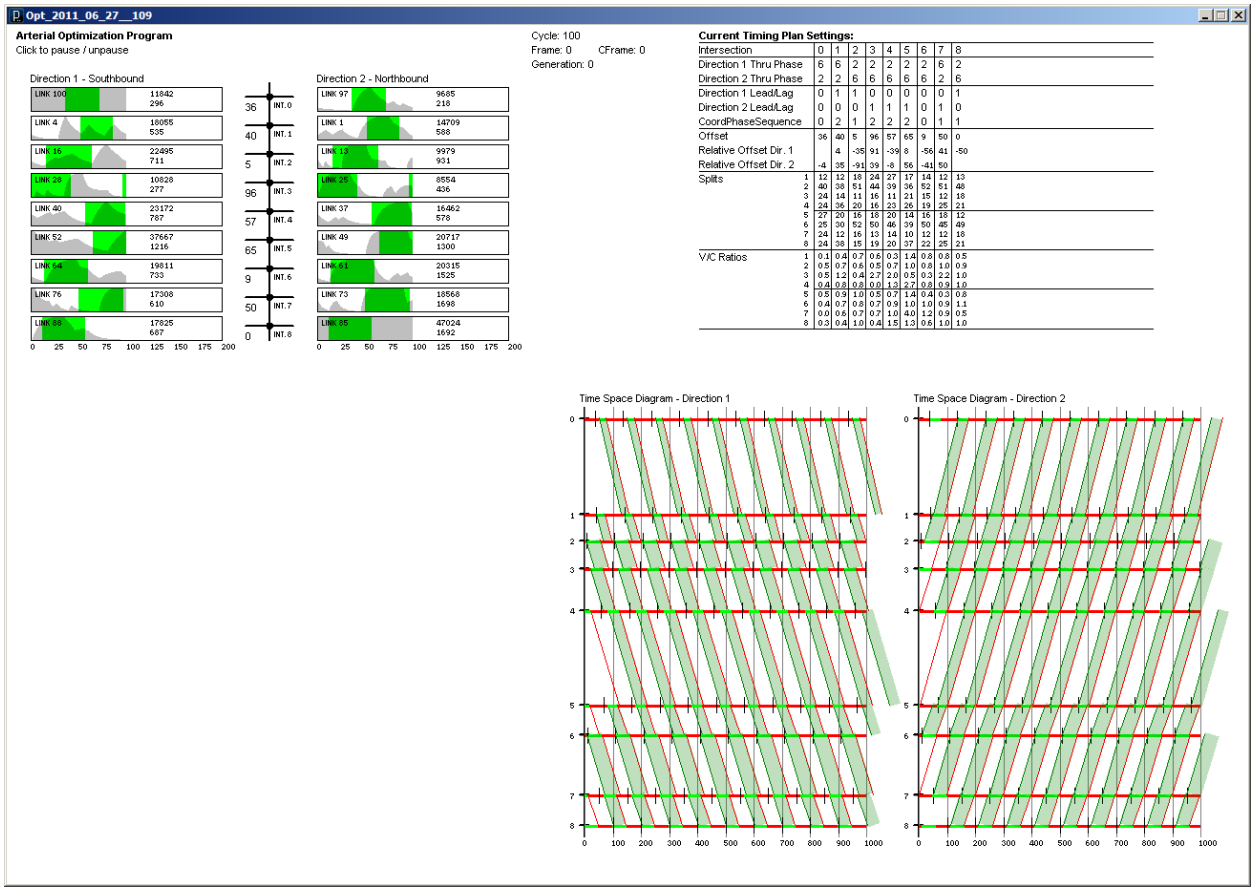


Figure 5: View of prototype software for arterial timing plan optimization.

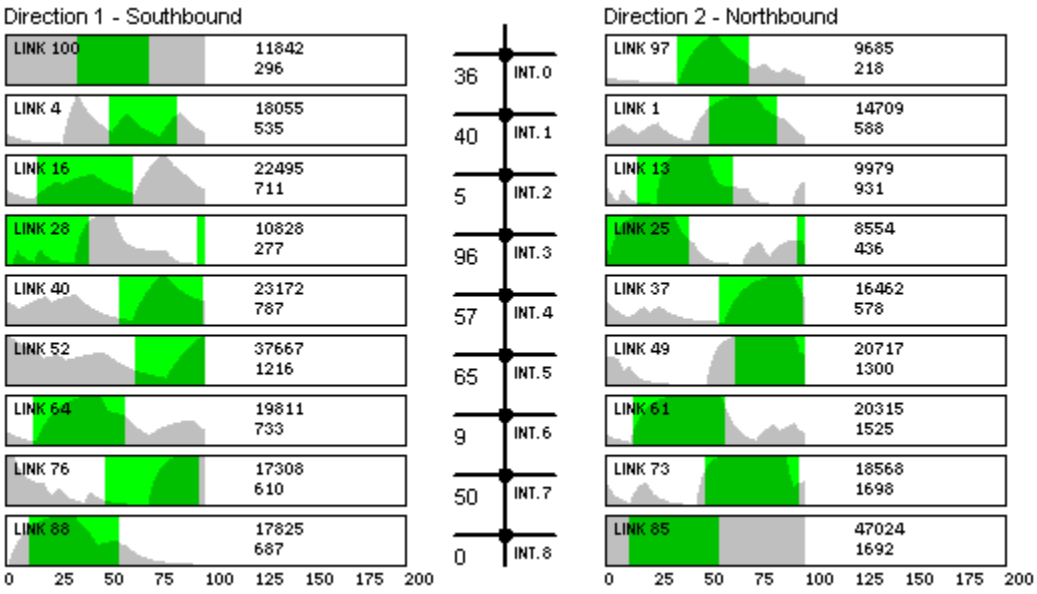


Figure 6: Abstracted flow profile view for SR 37.

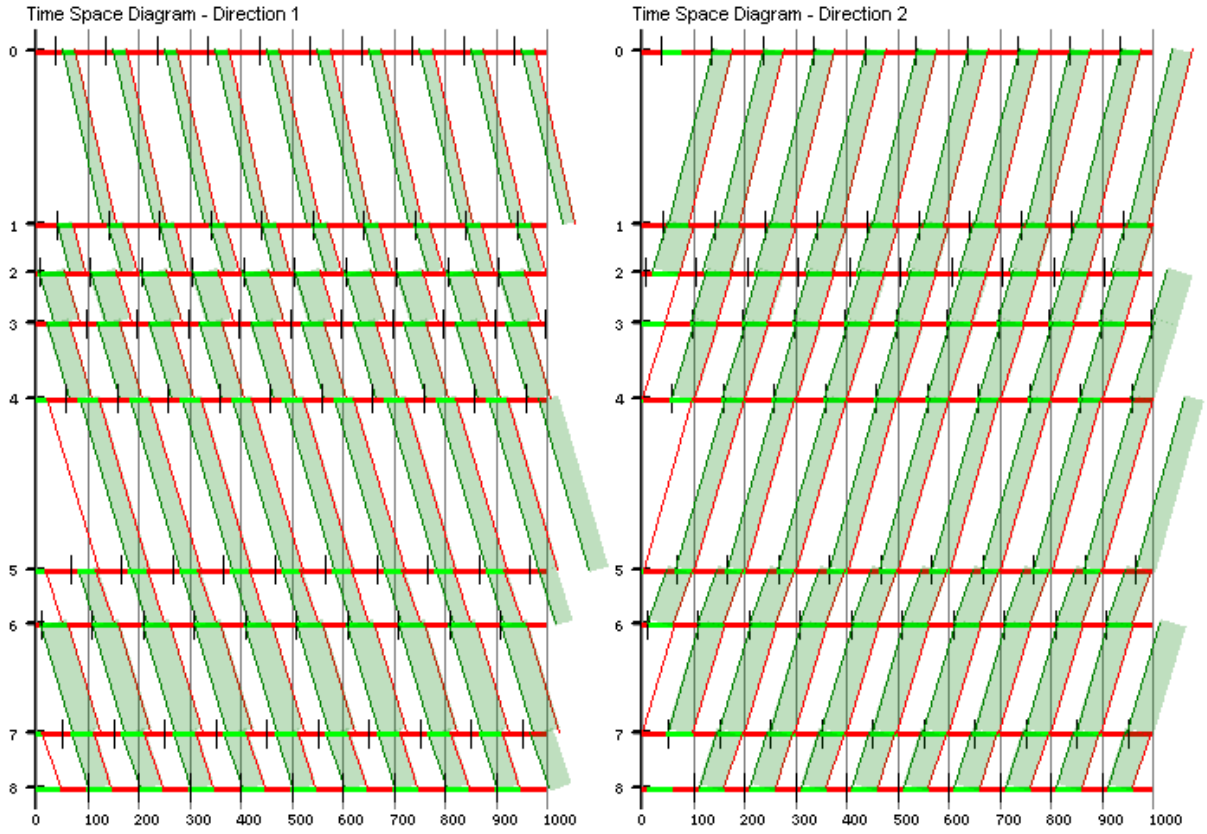


Figure 7: Time-space diagram view for SR 37.

Real-World Study Network

The methodology was applied for the SR 37 signal system in Noblesville, IN. Figure 8 shows a map of the system with the intersection numbering scheme used in TSM 3409.

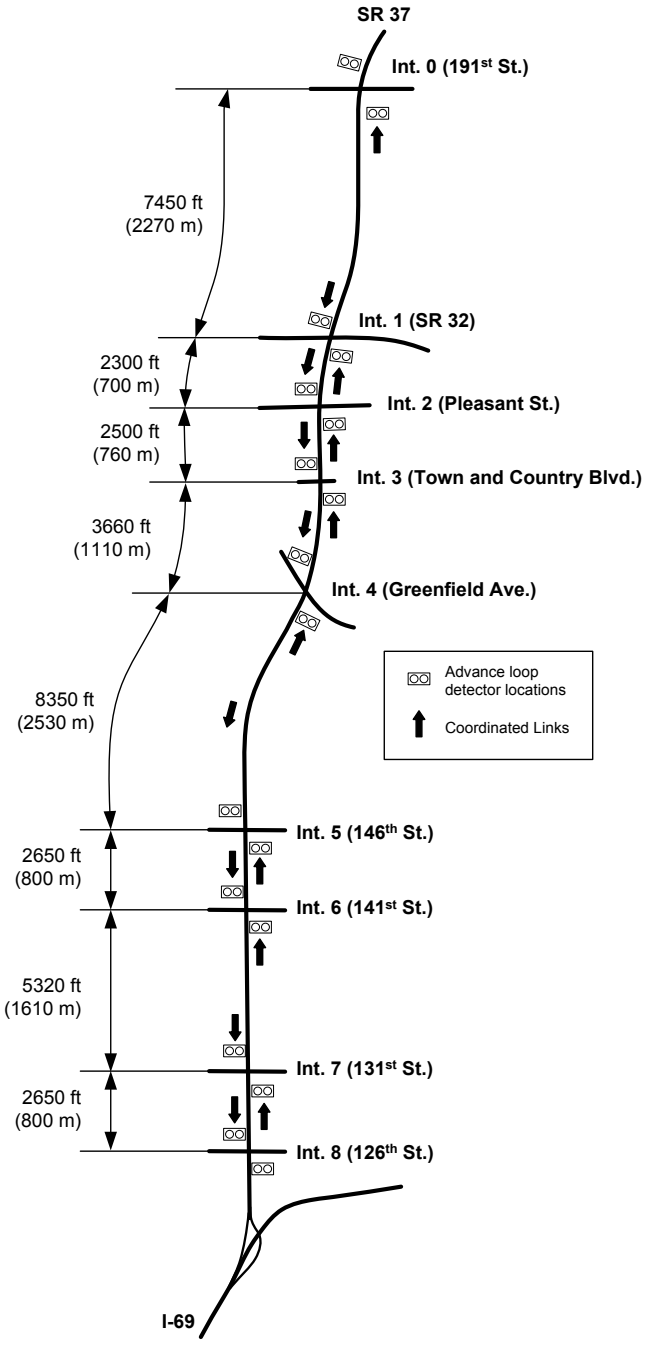


Figure 8: SR 37 in Noblesville, IN.

Comparison of Modeled and Field Observed Flow Profiles

To validate that the TSM 3409 model is capable of reproducing field conditions reasonably accurately, high-resolution signal event data was collected and used in the model as follows:

- The platoon dispersion parameters *T* and *F* were obtained through field observations using the methodology described by Day *et al*. (3)[2]. The *F* parameter is related to the α and β dispersion parameters by the following equation:

$$F = \frac{1}{1 + (\alpha\beta) \cdot T} \qquad \text{Equation 23}$$

- Field-observed volumes were used as the volume quantity *v* for each link in the system.
- Controller timing plans were used in TSM 3409 as the baseline timing plan and used to generate the test state.

Figure 9 compares the field observed flow profiles with the results of TSM 3409. With a few exceptions, the location of most of the primary coordinated platoons are located within the same part of the cycle as observed in the field, indicating that the model provides a reasonable estimation of the expected traffic state under observed conditions. A few notable exceptions are discussed below:

- TSM 3409 does not model right turns on red or the impact of phase actuation, which is why some platoons appearing in the model do not have corresponding platoons in the observed profiles. An extreme example is southbound at Int. 1, where the dominant upstream flow is the right turn; the observed inflow appears nearly random.
- The northbound approach at Int. 8 is actually random arrivals, but the queues at this particular location extend well beyond the detector during the analysis period (1500-1900), which makes its observed flow profile inaccurate. The uniform arrival distribution in the TSM 3409 flow profile results from the fact that no upstream links are defined.

[2] See Appendix E.

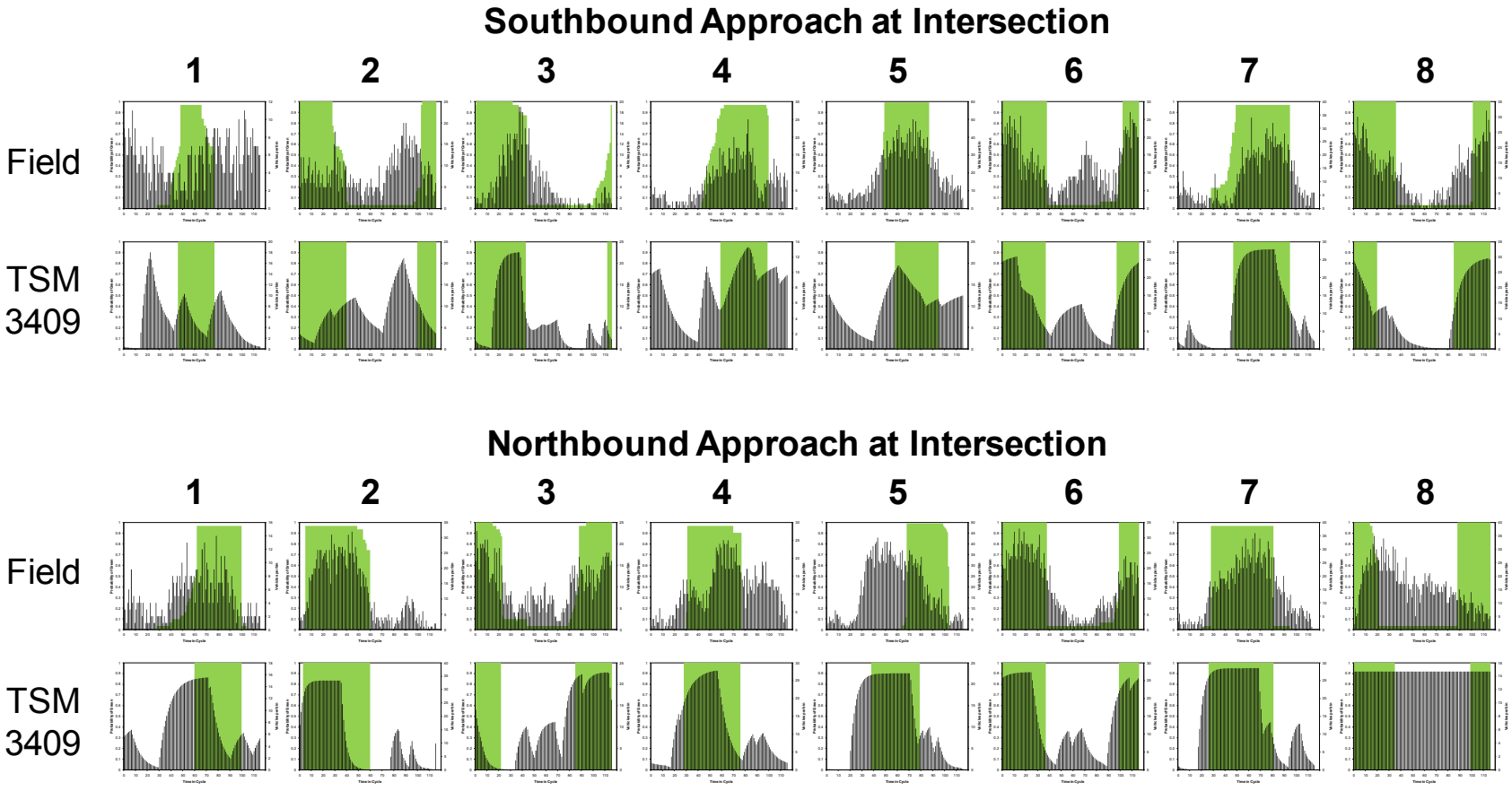


Figure 9: Comparison of field observed and macroscopic simulation flow profiles.
Field flow profiles obtained from data collected on March 16, 2011, 15:00-19:00.

OPTIMIZATION PROCEDURE OVERVIEW

To explore the optimization of cycle length, several optimization algorithms were coded that made use of the previously described model (TSM 3409) to estimate the network state under varying timing plan configurations. In this study, the settings of *cycle length, offset,* and *phase sequence* are explored. Figure 10 presents a diagram explaining the hierarchy of routines within the program (which execute different optimization algorithms) as applied in TSM 3409.

The initialization procedure loads all of the relevant data (link definitions) into the necessary variables and kicks off the main loop of the program. Each iteration of that loop runs one "generation" in a genetic algorithm (GA) procedure, wherein a large population of alternative signal timing patterns are tested. This testing happens within the "CalculateAll" procedure, which updates the network state variables to estimate the performance of the different timing plans. The best-known solution for the generation is then displayed at the end of each step.

In addition to the GA procedure, the *signal offsets* belonging to each member of the GA population may be optimized by one of two sub-steps: a hill climbing algorithm, as used in TRANSYT, or a "link pivot" algorithm (4). These two sub-steps themselves require numerous testing of alternative signal timing plans. Executing these by changing the network state variables would require a considerable amount of computation time (since the algorithm is executed for each member of the GA population). To lessen the computational burden, both the hill climbing and link pivot algorithm use an abstraction layer to store only the *relevant* link state information (i.e., the coordinated links). This excludes all the links that would be unaffected by the offset optimization process, but would have to be refreshed anyway during the "Calculate All" procedure. The optimal offsets are then calculated by systematically adjusting the smaller subset of link states in the abstraction layer. Once the optimized offsets are obtained, they are plugged into the global network state variables as the final step. This approach considerably reduces the computation time of the program.

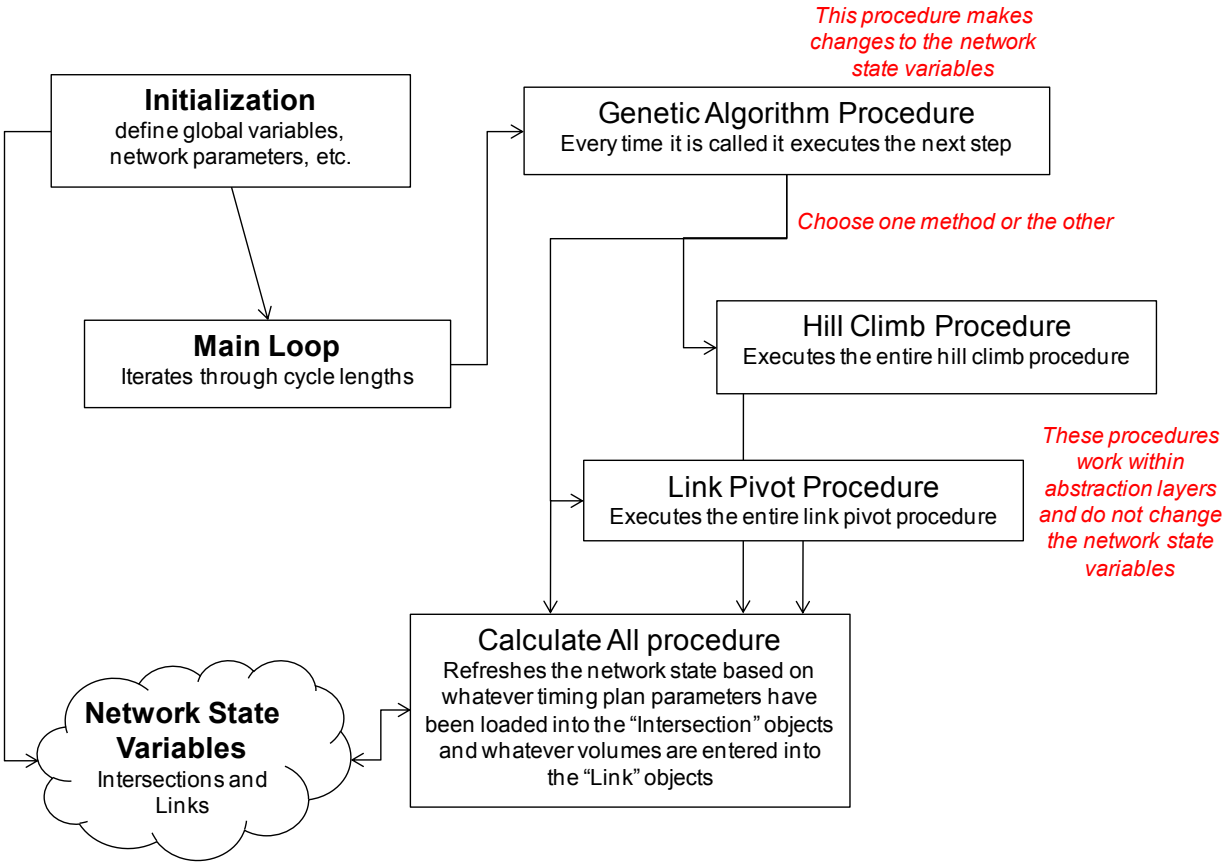


Figure 10: Overview of the program hierarchy.

Genetic Algorithm

Genetic algorithms (GAs) (5) have been used in numerous signal timing optimization studies (*6,7,8,9,10,11,12,13,14*). Figure 11 shows a diagram of the implementation of a GA in this work. The algorithm begins by generating a population of p members (scenarios) with randomly selected offsets and phase sequences[3]. The parameters are encoded into a binary number string called a "chromosome." The "fitness" of each member is then calculated (by testing its performance with the TSM 3409 model) proportionally to a performance measure. In this study, fitness is inversely proportional to *PI*. The algorithm then generates a number of successive generations $k = 1, 2, \ldots$ in an iterative process where each stage is called a "generation." During each generation, the following rules are applied:

- *Elitism*. The member with the highest fitness is automatically carried over into the next generation.
- *Selection.* Two "parent" members are chosen from the population by "roulette wheel" selection, where the probability of selection is proportional to fitness.
- *Crossover.* If a random number between 0 and 1 (`rand()`) is greater than the probability of crossover $P_{\text{crossover}}$, then the two chromosomes of the parents are merged by taking a portion of each chromosome and dividing them at a randomly selected midpoint location, then taking the complementary pieces and concatenating them. If crossover is not initiated, then one of the parent chromosomes (selected at random) is copied over in its entirety.
- *Mutation.* If a random number is greater than the probability of mutation P_{mutation}, then
- *Subprocess optimization.* In this implementation, the offsets used in the member of the population are optimized by a subprocess (either hill climbing or link pivoting).
- The fitness of the new member of the next "generation" is then calculated.

This process gradually produces better solutions as new generations are continually generated from the previous ones. In this implementation, the procedure is halted after a number of generations called the "quitting parameter" *QP*.

---

[3] In this study, constant-percentage splits are used in all scenarios that are externally optimized for the relative volumes.

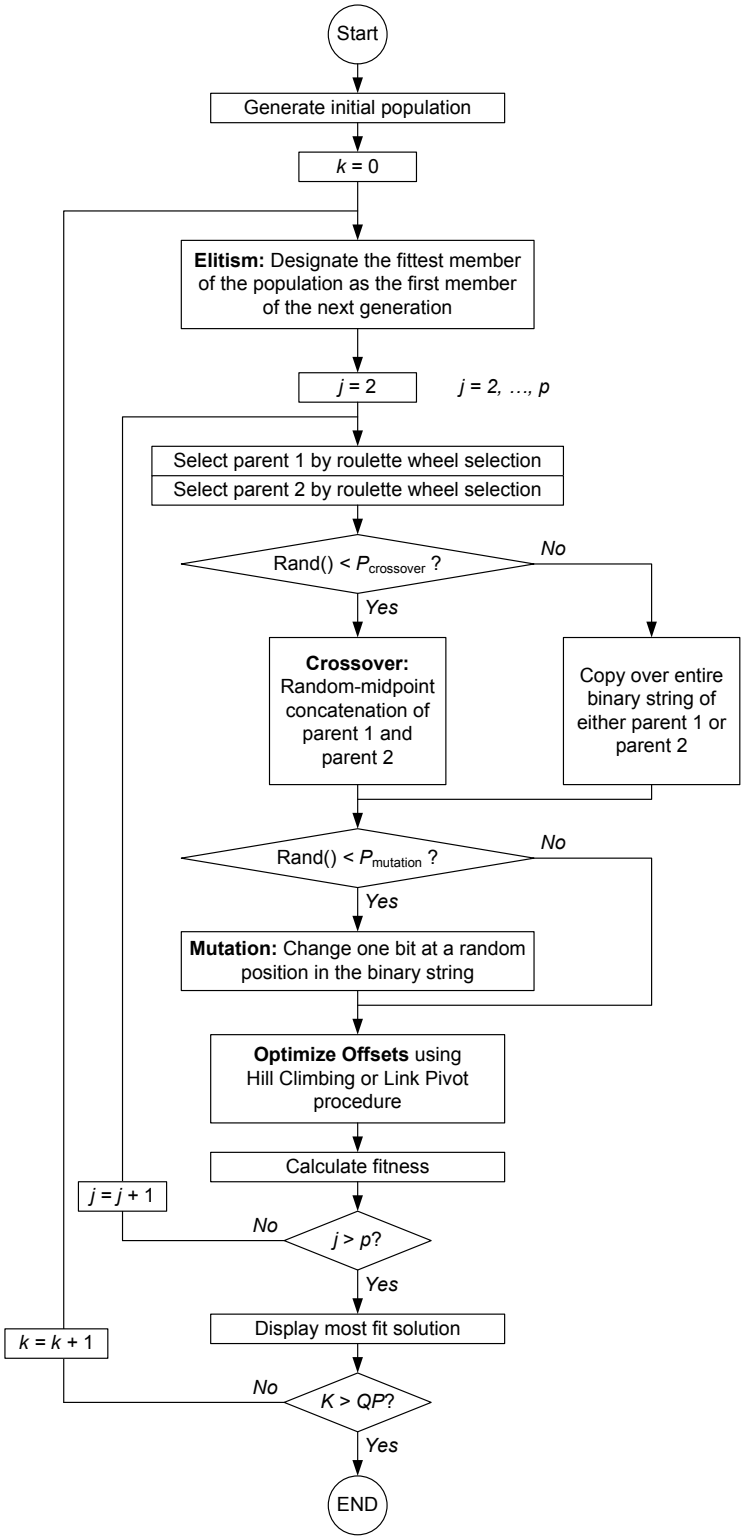


Figure 11: Genetic algorithm with an embedded offset optimization process.

Hill Climbing Algorithm

Hill climbing (HC) is a basic optimization strategy that was used in the first implementation of TRANSYT to optimize offsets (2). In this study, HC was used as one of the two optional sub-processes in the main GA optimization algorithm. The GA procedure is controlled mainly by random processes, in which none of the parameters are exposed to any trial-and-error refinements. Including a hill-climbing procedure introduces this type of optimization strategy into the process (*14*).

Figure 12 presents an illustration of how the HC algorithm works. The procedure is launched at the end of the GA procedure in Figure 11. The algorithm first calculates the initial system delay $d_{(base)}$, and then begins an iterative process; on each iteration, the algorithm moves to intersections $i = 1, \ldots n$ in the system, and tests out a series of $j = 1, \ldots, h$ adjustments, or "hill climb increments" on those offsets. The increments are stored in a vector **H**. In this study, we used the default TRANSYT increments of {-45, -15, -5, -1, 0, 1, 5, 15, 45}. The increment H_j is implemented at intersection i and the trial delay $d_{(trial)}$ is calculated. This is compared to the delay value prior to making the adjustment [$d_{(i)}$]. If delay has been reduced, the adjustment is saved as the best known adjustment; otherwise it is discarded. After each intersection has been "visited" by the algorithm in the iteration, it starts over. At the end of every iteration, if no further improvement has been achieved (i.e., if $d_{(new)} \not\leq d_{(base)}$), then the algorithm stops.

Computational requirements of HC are limited by the fact that, during each trial implementation of a hill climb increment, the state variables of only those links approaching or exiting an intersection need to be adjusted. For an arterial network, this means two arriving links and two departing links.

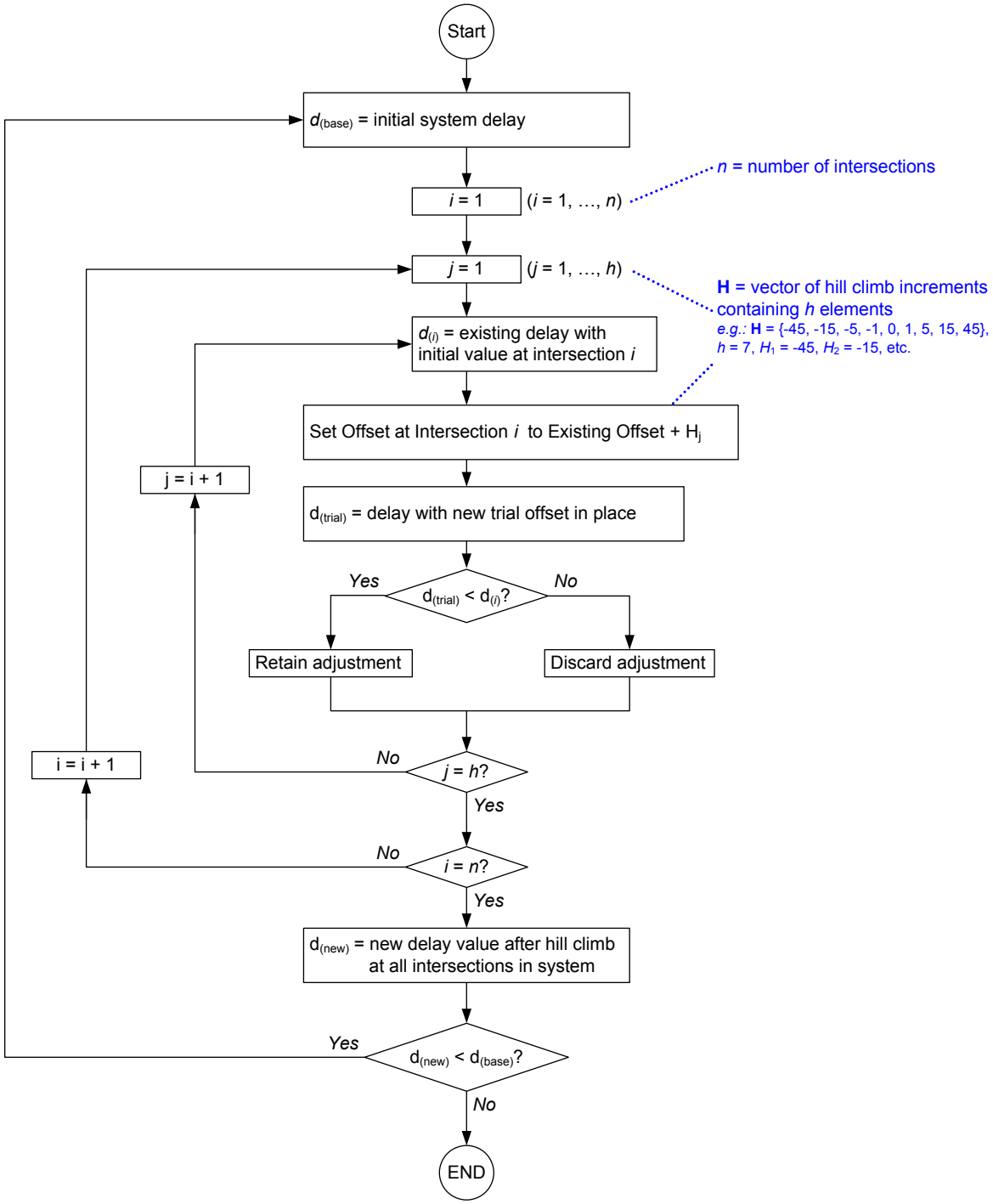


Figure 12: Hill climbing algorithm.

Link Pivot Algorithm

For network topologies that do not contain closed loops, it is possible to find the global optimal offset by systematically adjusting offsets of intersections in portions of the system, while maintaining the relative offsets within the optimized group constant. This is essentially the procedure of the Combination Method (*15,16*), a 1960s optimization software package developed in the UK, with several improvements by successive researchers (*17,18,19,20*). In NCHRP 3-79A, this method was formulated for arterials (4) and called "link pivoting" (LP).

Figure 13 illustrates how the algorithm works. For an arterial, first we determine the direction that the algorithm is to proceed on the arterial (from intersection j_0 to j_{max})[4]. The algorithm then "walks" the arterial in that order. At each intersection, it does an exhaustive search of the possible offset adjustments ($\delta = 0, \ldots, C - 1$), and applies the best known δ value to that intersection's offset. Now, when the algorithm moves to intersection $j + 1$, during each test of successive δ values, it applies the *same adjustment* to all intersections from j_0 to j, which effectively maintains the same offset in that group of intersections. In this way, the algorithm "pivots" about each link in the system (hence the name "link pivot") until it has fully optimized the entire network. Note, lastly, that the δ value at the final intersection j_{max} can be set to any value without affecting the overall system performance; this means that it can be set to any desired value to achieve any particular offset at any particular intersection. In any group of intersection offsets, *one* offset must be arbitrarily set to *some* value in order to serve as a reference point for the others. Customarily, an offset of zero is usually established at the master intersection.

The LP algorithm is computationally very similar to HC. During the trial of every δ value, the state variables of only *two links* need to be adjusted (those on the link currently being "pivoted" around). While generally more trials are needed because the δ values cover the full range of cycle lengths, HC is an iterative process, so the total number of computations is similar (4). Both are fast algorithms; HC is applicable to all networks in general, while LP works well for those without closed loops.

---

[4] For non-arterial networks, a priority-tree approach could be used (e.g., *18*).

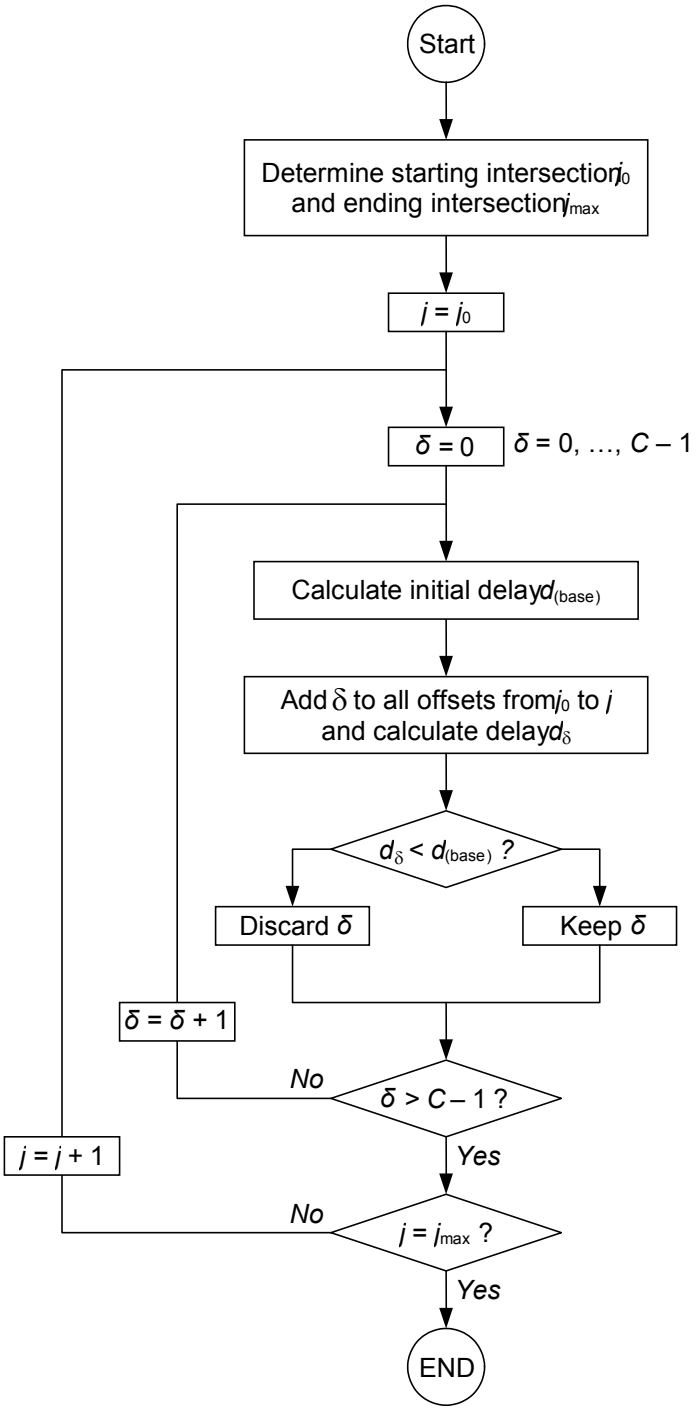


Figure 13: Link pivot algorithm.

RESULTS

To test various algorithms in the optimization program and investigate cycle length optimization, we carried out several "sweeps" of the cycle length (from C = 70 s to C = 140 s) for the following alternative algorithm configurations:

1. *Monte Carlo simulation*. For each cycle length, 1000 random offset / sequence scenarios were generated. No other optimization was executed for those scenarios.
2. *GA only*. A genetic algorithm was executed without any embedded offset optimization process. A population size of p = 10 was used. QP = 40 generations were allowed to pass before the algorithm was forced to quit. GA parameters of $P_{\text{crossover}} = 0.7$ and $P_{\text{mutation}} = 0.2$ were used. For each cycle length, the GA was initialized 20 times to mitigate the impact of the initial randomly generated conditions in evaluating the algorithm performance.
3. *Hybrid GA + HC*. The same parameters as for "GA only" were used. Additionally, offsets were optimized for each GA solution using HC.
4. *Hybrid GA + LP*. The same parameters as for "GA only" were used. Additionally, offsets were optimized for each GA solution using LP.

The optimization objective in each case was to minimize the performance index (*PI*). The GA fitness value as well as the HC / LP objective functions were all based on *PI*.

Figure 14 shows the results of this procedure. This graph shows four series of box-whisker plots corresponding to the four algorithm configurations listed above. For each cycle length, the "whiskers" indicate the min/max values, and the "box" shows the 25th percentile, median, and 75th percentile of 1000 random scenarios from Monte Carlo simulation, and 20 separate runs from GA and the two Hybrid-GA algorithm configurations.

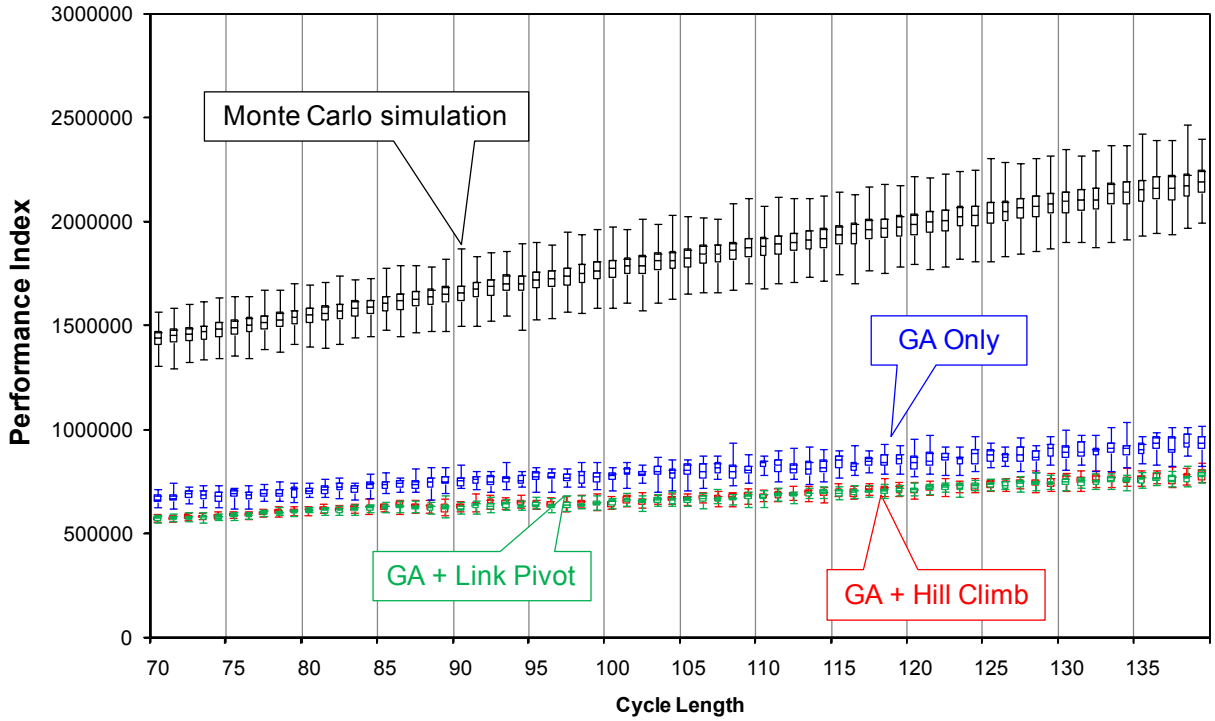


Figure 14: Comparison of algorithm results for the SR 37 test network.
The box-whisker plots show distributions from 20 runs per cycle length for the GA and hybrid-GA algorithms, and 1000 runs for Monte Carlo simulation.

Figure 14 provides several results with regard to algorithm performance as well as cycle length optimization:

- *Algorithm performance*. The Monte Carlo simulation results show the performance of typical offset/sequence configurations for each cycle length (from random selection). The GA optimization substantially improves upon these results, reducing *PI* by a factor of approximately 2 in most cases; the range of results from 20 separate GA runs is completely separated from the range of Monte Carlo simulations. The hybrid-GA results improve on the optimality of the solutions and are similarly fully below the range of the GA-only runs. The reliability of the hybrid-GA results were improved, as shown by the tighter distribution ranges compared to GA-only. There was little difference between LP and HC in terms of performance.
- *Cycle length optimization*. For all algorithms, *PI* is found to monotonically increase with cycle length. It was speculated that a "resonant cycle length" (*21,22*) might be discovered in the system that would represent a local *PI* minimum with respect to cycle length. However, this was not observed in this particular network. From this, it is possible to conclude that the conventional method of selecting a network cycle length is likely the best method: namely, to use the minimum cycle length that satisfies the capacity needs of all signals in the coordinated group.
- Additionally, all three optimization algorithms seem about equally capable of reducing the *slope* of the monotonic increase. That is, the difference in delay between a 120-sec cycle versus a 100-sec cycle is less for the results of the optimized settings versus Monte Carlo simulation.

CONCLUSIONS

This paper presented a formulation of a macroscopic traffic model, TSM 3409, that is used as a basis for signal timing plan optimization. A series of algorithms for optimization of offset, cycle length, and sequence are presented. A genetic algorithm is used as a basis of optimizing phase sequence and a population of random offsets. Subroutines for offset optimization are included in a hill climbing or link pivoting algorithm. Compared to Monte Carlo simulation (multiple sequential random parameters scenarios), the genetic algorithm is able to optimize the signal timing plan, reducing the median solution performance index by greater than a factor of 2. Combining this genetic algorithm process with a subroutine algorithm increases the performance of the optimization process further. The performance of link pivot and hill climbing were very similar in this context. Future work would consist of testing the strategies in field implementation, augmenting the optimization procedure to incorporate additional signal timing parameters, and testing optimization outcomes with alternative objective functions.

ACKNOWLEDGMENTS

This work was supported by the Joint Transportation Research Program administered by the Indiana Department of Transportation and Purdue University. The contents of this paper reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein, and do not necessarily reflect the official views or policies of the sponsoring organizations. These contents do not constitute a standard, specification, or regulation.

REFERENCES

1. Reas, C. and Fry, B. *Getting Started with Processing.* O'Reilly Media Inc., Sebastopol, CA, 2010.

2. Robertson, D.I. *TRANSYT: a traffic network study tool*. Report No. LR 253, Road Research Laboratory, Crowthorne, Berkshire, England, 1969.

3. C.M. Day and D.M. Bullock. "Using Field Data to Improve Model Accuracy: Application of the Robertson Dispersion Model to High-Resolution Signal Event Data." Submitted to *Transportation Research Record* August 1, 2011, Paper No. 12-0061, under review.

4. C.M. Day and D.M. Bullock. "Computational Efficiency of Alternative Arterial Offset Optimization Algorithms." *Transportation Research Record*, Paper No. 11-0181, accepted for publication, in press.

5. Holland, J. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

6. Foy, M.D., Benekohal, R.F., and Goldberg, D.E. Signal timing determination using genetic algorithms. *Transportation Research Record No. 1365*, Transportation Research Board of the National Academies, Washington, DC, pp. 108-115, 1992.

7. Memon, G.Q., and Bullen, A.G.R. Multivariate optimization strategies for real-time traffic control signals. *Transportation Research Record No. 1554*, Transportation Research Board of the National Academies, Washington, DC, pp. 36-42, 1996.

8. Oda, T., Otokita, T., Tsugui, T., Kohno, M., and Mashiyama, Y. Optimization of signal control parameters using a genetic algorithm. *Proc., 3rd Annual World Congress on Intelligent Transportation Systems*, Orlando, FL, 1996.

9. Park, B. *Development of Genetic Algorithm-based Signal Optimization Program for Oversaturated Intersections.* PhD Thesis, Texas A&M University, 1998.

10. Yun, I. and Park, B. *Stochastic Optimization Method for Coordinated Actuated Signal Systems*. Report No. UVACTS-15-0-102, Center for Transportation Studies, Univ. of Virginia, Charlottesville, VA, 2004.

11. Stevanovic, J., Stevanovic, A., Martin, P.T., and Bauer, T. Stochastic optimization of traffic control and transit priority settings in VISSIM. *Transportation Research Part C*, Vol. 16, pp. 332-349, 2008.

12. Kesur, K.B. Advances in genetic algorithm optimization of traffic signals. *ASCE Journal of Transportation Engineering*, Vol. 135, pp. 160-173, 2009.

13. Park, B., Messer, C.J., and Urbanik, T. Enhanced genetic algorithm for signal-timing optimization of oversaturated intersections. *Transportation Research Record No. 1727*, Transportation Research Board of the National Academies, Washington, DC, pp. 32-41, 2000.

14. Ceylan, H. Developing combined genetic algorithm—hill-climbing optimization method for area traffic control. *Journal of Transportation Engineering*, ASCE, Vol. 132, pp. 663-671, 2006.

15. Hillier, J.A. "Appendix to Glasgow's Experiment in Area Traffic Control." *Traffic Engineering and Control*, Vol. 7, pp. 569-571, 1965.

16. Huddart, K.W. and Turner, E.D. Traffic signal progressions – GLC combination method. *Traffic Engineering and Control*, Vol. 11, pp. 320-322, 1969.

17. Gartner, N. Optimal synchronization of traffic signal networks by dynamic programming. In *Proc. Traffic Flow and Transportation: 5th International Symposium on the Theory of Traffic Flow and Transportation,* ed. G.F. Newell, Elsevier, New York, pp. 281-295, 1971.

18. Inose, H, H. Fujisaki, and T. Hamada. Theory of road-traffic control based on macroscopic traffic model. *Electronics Letters,* Vol. 3, pp. 385-386, 1967.

19. Gartner, N.H. and Little, J.D.C. Generalized combination method for area traffic control. *Highway Research Record No. 531*, Transportation Research Board of the National Academies, Washington, DC, pp. 58-69, 1975.

20. Improta, G. and Sforza, A. Optimal offsets for traffic signal systems in urban networks. *Transportation Research Part B*, Vol. 16, pp. 143-161, 1982.

21. Koshi, M. Cycle time optimization in traffic signal coordination. *Transportation Research Part A*, Vol. 23, pp. 29-34, 1989.

22. Shelby, S.G., Bullock, D.M., and Gettman, D. Resonant cycles in traffic signal control. *Transportation Research Record No. 1925*, pp. 215-226, 2005.