Developing Combined Genetic Algorithm—Hill-Climbing Optimization Method for Area Traffic Control

Halim Ceylan[1]

Abstract: This study develops a genetic algorithm with TRANSYT hill-climbing optimization routine, referred to as GATHIC, and proposes a *method* for decreasing the search space, referred to as ADESS, to find optimal or near-optimal signal timings for area traffic control (ATC). The ADESS with GATHIC model is an algorithm, which solves the ATC problem to optimize signal timings for all signal controlled junctions by taking into account coordination effects. The flowchart of the proposed model with ADESS algorithm is correspondingly given. The GATHIC is applied to a well-known road network in literature for fixed sets of demand. Results showed that the GATHIC is better in signal timing optimization in terms of optimal values of timings and performance index when it is compared with TRANSYT, but it is computationally demanding due to the inclusion of the hill-climbing method into the model. This deficiency may be removed by introducing the ADESS algorithm. The GATHIC model is also tested for 10% increased and decreased values of demand from a base demand.

DOI: 10.1061/(ASCE)0733-947X(2006)132:8(663)

CE Database subject headings: Traffic control; Traffic delay; Optimization models; Algorithms.

Introduction

Traffic signal control is a multiobjective optimization encompassing *delay, queuing, pollution, fuel consumption*, and *traffic throughput,* combined into a *network performance index* (Akcelik 1981). It can be either stage based (e.g., TRANSYT) or group based (e.g., SIGSIGN) (Silcock and Sang 1990; Allsop 1992). Signal optimization applies to several decision variables, such as green time, cycle length, stage sequence, and offset. The optimization of signal timing for an isolated junction is relatively straightforward, but optimizing the timing in dense networks where the distances between the intersections are too small to dissipate the platoons of traffic is a difficult task. The difficulty comes from the complexity of the signal *coordination*. Optimization of signal timings is well established at individual junctions (Heydecker and Dudgeon 1987; Gallivan and Heydecker 1988; Allsop 1992; Heydecker 1992), but optimization of timings in coordinated signalized networks requires further research due to the offsets and common network cycle time requirements. The reason for further research is at least twofold: One is the nonconvexity of the optimization problem when it is formulated as a mathematical program, and the second is that there is no feasible constraint for the start of the green (i.e., signal offsets), thus making the problem an unconstraint optimization problem.

Among the current optimization models for area traffic control, TRANSYT is one of the most useful network study software tools for optimizing splits, offsets, and stage ordering, and is also the most widely used program of its type. TRANSYT was developed by Transportation and Road Research Laboratory (Robertson 1969) and is a stage-based optimization program. Its main features are the cyclic flow profile and platoon dispersion models, and the hill-climbing algorithm. The model consists of two main parts: A traffic flow model and a signal timing optimizer. The traffic model (Vincent et al. 1980) is a deterministic mesoscopic time-scan simulation. It simulates traffic in a network of signalized intersections to produce a cyclic flow profile of arrivals at each intersection that is used to compute a performance index (PI) for a given signal timing and staging plan. The performance index is defined as the sum for all signal-controlled traffic streams of a weighted linear combination of estimated delay and number of stops per unit time and is used to measure the overall cost of traffic congestion associated with the traffic control plan.

The optimization procedure in TRANSYT is based on an iterative search technique, known as “hill-climbing” (HC), which basically searches for the best signal timings by a trial and error method. The HC operates on two signal setting variables: The offset, which affects the coordination between junctions, and the stage start and end times. Its optimization process can be described as follows.

First, TRANSYT calculates the performance index for an initial set of signal timings, in which all signal operational constraints are satisfied for safety considerations. Next, one of the signal control variables is changed by a predetermined number of steps and the corresponding value of performance index is calculated. If the calculated value for the performance index decreases, which means that the system performance is improved when the signal setting variables changed in the prespecified, the signal setting variable is then altered in the same direction by the same number of steps until a minimum value of the performance index is obtained. On the other hand, if the calculated value of the performance index does not decrease, which shows that the system performance is not improved as the signal setting variable is changed in the chosen direction, the same variable is altered in

[1]Associate Professor, Ins. Muh. Bol. Muh. Fak. Pamukkale Univ., Denizli, 20017, Turkey. E-mail: halimc@pamukkale.edu.tr

Note. Discussion open until January 1, 2007. Separate discussions must be submitted for individual papers. To extend the closing date by one month, a written request must be filed with the ASCE Managing Editor. The manuscript for this paper was submitted for review and possible publication on May 10, 2004; approved on December 9, 2005. This paper is part of the ***Journal of Transportation Engineering***, Vol. 132, No. 8, August 1, 2006. ©ASCE, ISSN 0733-947X/2006/8-663–671/ $25.00.

the opposed direction by the same number of steps until a minimal value of the performance index is obtained. This process continues for each signal setting variable in the road network in turn. The steps by which the different variables are changed can be determined in advance (see, for details, Vincent et al. 1980).

The HC uses an iterative improvement technique that is applied to a single point in the search space. At each iteration, a new point is selected from the neighborhood of the current point. If the new point provides a better value of the objective function, the new point becomes the current point. The search process terminates when no further improvement is possible.

On the other hand, genetic algorithms (GAs) perform a multidirectional search by maintaining a population of potential solutions and encourages information exchange between these directions. GAs are a family of computational tools inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome string, such as data structure, and apply recombination operators to these structures so as to preserve critical information, and to produce a new population with the intent of generating strings which map to high function values. GAs utilize concepts derived from biology as they are based on Darwin's theory of survival of the fittest.

The present study includes an implementation of binary-coded GA. Local rules of interaction replace the traditional stochastic operations of *selection* and *crossover*. The evolution process is conducted locally with probabilistic transformation rules. Each site contains a binary bit description of the signal timings. The convergence characteristics of the GA are improved through the use of a *shuffle* mechanism that simply relocates members of the population to new sites within the search space at random after the mutation operations. This allows for new information to be introduced in the local neighborhood.

The GA works with the expression operation that is performed based on fitness evaluation. It searches the most fit members by minimizing the total network PI obtained by the TRANSYT traffic model.

Both the GA and HC optimization routines have advantages and disadvantages in terms of their ability to find a global solution and central processing unit (CPU) time. The problem in the HC process is the possibility of being trapped at bad local optima. The population-based search attempts to escape from falling into bad local optima. Hence, good optimization routine may be obtained by mathematically combining GA and HC techniques in order to remove being trapped in bad local optima or to locate a good global optimum. In addition, there is no study to decrease the search space for the GA and HC so that the performance of the TRANSYT model may be improved in terms of CPU time.

The TRANSYT-7F releases 10 features a separate GA optimization of cycle length, stage sequence, splits, and offsets. The phasing sequence optimization by TRANSYT-7F is also available not only for the entire network and but also for each individual intersection. But any of the versions of this means that TRANSYT may not explicitly combine the GA and, the HC method to optimize all signal timing variables at the area traffic control problem.

Heydecker (1996) proposed a decomposition approach to optimize signal timings at individual and at network levels based on the group-based variables. In this approach, two levels of optimization were carried out alternatively until certain convergence criteria were satisfied. Considerable computational advantages were obtained. It was however pointed out that each level of optimization could only produce suboptimal results and hence there was no guarantee of convergence. Each level of optimization was suboptimal because the effects of the coordination between adjacent junctions were not taken into account.

Wong (1996) proposed an optimization of signal timings for area traffic control using group-based control variables. In this case, the TRANSYT performance index is considered as a function of the group-based control variables, cycle time, start and duration of green time. The signal timing optimization problem is further formulated as nonlinear mathematical programs, in which the performance index of a network is minimized subject to certain constraints. The problem is solved using integer-programming methods. A trial network from Leicestershire was used to demonstrate the effectiveness of the proposed method. About 10% improvements in the optimal performance indices over the stage-based method in TRANSYT were obtained. However, it was reported that obtaining the derivations of the performance index for each of the control variable was mathematically difficult. In addition, random offset calculation was proposed to locate better local minimum, but it requires much longer computational time.

In relation to the network common cycle time, the selection of the best cycle times for each node within a network is a complex and, as yet, unsolved optimization problem. In TRANSYT, selection of common cycle time based on PI of individual intersection obtained through a quick optimization of splits, then followed by a more detailed optimization. It is one of the advantages of the GA that the common cycle time can be considered as a control variable in the optimization process, so that a common cycle time which works in the most harmonious way with the coordination patterns of a network can be determined.

A difficulty could arise in traditional methods because of the use of a common cycle time for all signal-controlled junctions in the network. Imposing common cycle time constraint to all junctions might lead to the suboptimality of the sequence and interstages. Optimality of the sequences and interstage structures is controlled by reoptimizing each junction individually with the cycle time constraint to be equal to the common cycle time. If this leads to the selection of new sequences or interstage structures, the common network cycle time can then be reoptimized with the new data and process repeated (Heydecker 1996). However, the possible need to revise the sequences and interstage structures several times adds to the complexity of the problem. Thus, in order to optimize common cycle time at network level and then adjust the individual junction's timings, there is a need for a heuristic search method, such as a combined GA and HC method.

The common cycle time variable can only be increased, decreased or remain unchanged. It is well known that the longer the cycle time, the more the capacity, but the more delay. Moreover, there are network implications of selecting a cycle that provide sufficient capacity at all intersections, but may impose undue large delays at many intersections. Thus, there is a need for new search methods taking into account the network common cycle time as a decision variable while simultaneously optimizing individual junction's signal timings. For this purpose, a one-step optimization heuristic, combining GA with HC, is developed. For each change of cycle time, the GA with HC is performed, called GATHIC with ADESS, to find the minimum PI in the signal-controlled network. Although the TRANSYT-7F may perform a common cycle length evaluation with GA, but there is no optimization routine for the HC procedure and no algorithm for reducing the search space for the GA as mathematical program.

As a solution for above-mentioned problems, the **A**lgorithm for **DE**creased **S**earch **S**pace (ADESS) is introduced. It provides a common cycle time, decreased, increased, or fixed, and reports an optimum cycle time with GATHIC. With this approach, the disadvantage of CPU time of the GA search space may also be relaxed. This study proposes a combined GA with the HC method, referred to as GATHIC, and proposes an algorithm to reduce the search space for the GA, referred to as ADESS.

GAs

GAs are a family of adaptive search procedures that are loosely based on models of genetic changes in a population of individuals. The paradigms of analysis and design based on the principles of biological evolution have been around since the 1960s. GAs were elucidated by Goldberg (1989) while Gen and Cheng (1997) later attracted the growing interest of optimization problems. The main advantage of GAs is their ability to use accumulating information about initially unknown search space in order to bias subsequent searches into useful subspaces. GAs differ from conventional nonlinear optimization techniques in that they search by maintaining a population of solutions from which better solutions are created rather than making incremental changes to a single solution to the problem.

GA Operators

The key feature of a GA is the manipulation of a population whose individuals are characterized by a chromosome. This chromosome can be coded as bits of given length, l, with each string representing a feasible solution to the optimization problem. A chromosome is further composed of strings of symbols called bits. Each bit is attached to a position within the string representing the chromosome to which it belongs. If, for example, the strings are binary, then each bit can take any value of 0 and 1. The link between the GA and the problem at hand is provided by the fitness function (F). The F establishes a mapping from the chromosomes to some set of real numbers. The greater the F value, the better the adaptation of the individual.

The procedure is generative. It makes use of three main operators; Reproduction, crossover, and mutation. Each generation of a GA consists of a new population produced from the previous generation. The number of individuals in a population is assigned feature value to their chromosomes, where the assignment can be either deterministic or random.

Reproduction is a process that selects the fittest chromosomes according to some selection operator. One example of a selection operator is the tournament selection (Goldberg and Deb 1991). This operator chooses the members that will be allowed to reproduce during the current generation according to the fitness values. Further manipulation is carried out by crossover and mutation operator before the replacement is actually done in the view of the next cycle.

Crossover provides a mechanism for the exchange of chromosomes between mated parents. Mated parents then create a child with a chromosome set that is some mix of the parents' chromosomes. For example, Parent No. 1 has chromosomes "abcde", while Parent No. 2 has chromosomes "ABCDE"; one possible chromosome set for the child is "abcDE"; where the position between the "c" and "D" chromosomes is the crossover point.

Mutation is an operator which produces spontaneous random changes in various chromosomes. A simple way to achieve mutation would be to alter one or more bits. The mutation operator serves a crucial role in genetic algorithms either by: (a) replacing genes lost from the population during the selection process or (b) providing genes that were not present in the initial population. The mutation process has a small probability that (after crossover) one or more of the child's chromosomes will be mutated, e.g., the child ends up with "abcDF". The purpose of this operator is to prevent the process from becoming trapped at a bad local optimum.

Apart from the main operators described so far, the other operator used in this study is the *elitism* operator, which is used to ensure that the chromosome of the best parent generated to date is carried forward unchanged into the next generation. After the population is generated, the GA checks to see if the best parent has been replicated; if not, then a random individual is chosen and the chromosome set of the best parent is mapped into that individual.

String Representation

The representation of the signal timings into binary strings requires determining the bit string length. The lower bound value corresponds to all zero digits (0000...), while the upper bound value corresponds to all one digits (1111...). The values between the lower and upper bound are linearly scaled and associated with corresponding binary strings. A design with multiple variables can be represented by stacking, head-to-tail, the strings in any given order. Stacking of signal setting variables can be represented as follows

$$\begin{array}{lcccc}
\text{Decision variables} \quad \psi= & |c| & |\theta_1,\theta_2,\ldots,\theta_{N_j}| & |\phi_1,\phi_2,\ldots,\phi_{N_j}| \\
\text{Mapping} \quad \downarrow & \downarrow & \downarrow & \downarrow \\
\text{Chromosome (string)}\mathbf{x} = & |01010101| & 01010111,\ldots,10101011| & 10101010,\ldots,01010010|
\end{array}$$

While dealing with binary string representation, one may need to use a large number of bits to represent the variables to high accuracy. Although a higher degree of precision can be obtained by increasing the string length, it is not always desirable because the computational cost of GAs also increase as the binary string gets longer. The higher number of bits will improve the performance of the GA algorithm due to the small step sizes as given in Eq. (2), but at a higher cost. The number of binary digits needed for an appropriate representation can be calculated from the following equation

$$2^{l_i} \geq (\psi_{i,\max} - \psi_{i,\min})/\Delta\psi_i + 1, \quad i = 1,2,3,\ldots,z \tag{1}$$

where $\Delta\psi_i$, can be calculated as

$$\Delta\psi_i = (\psi_{i,\max} - \psi_{i,\min})/(2^{l_i} - 1) \tag{2}$$

Suppose that the possible values of a cycle time, offset, and green times were 38, 14, 9, 34, 8, 30, 24, and 22 s. Then, the binary coding would be as shown below

c	θ_1	θ_2	θ_{31}	ϕ_1	ϕ_2	ϕ_3	ϕ_4
38	14	9	34	8	30	24	22
1111	0011	0001	1100	0000	1010	1001	0111

Then, $\Delta\psi_i$=2 s, and l=4, $\psi_{\min}$=8 and $\psi_{\max}$=38. So, if that section of chromosome reads "1100" then Θ_i=13, and ψ=34 s (fourth chromosome in the example).

Mapping from a binary string of design variables to real numbers is carried out in the following way:

$$\psi_i = \psi_{i,\min} + \Theta_i \Delta\psi_i, \quad i = 1,2,3,\ldots,z \tag{3}$$

Fitness Evaluation

With previous operations, a population is changed in form and characteristics, and represents a new generation. An iterative search after many generations of evolution leads the population to optimal or near-optimal signal timing variables. The GA works with the expression operation that are performed based on fitness evaluation. The fitness indicates the *goodness* of design, and therefore, the objective function is a logical choice for the fitness measure. The fitness function, $F(x)$, selected is

$$\underset{q=\text{fixed}}{\text{Max}}\ F(x) = 1/PI(\psi) \tag{4}$$

where the total network PI is formulated as

$$\underset{\psi,q=\text{fixed}}{\text{Min PI}} = \sum_{a\in\mathbf{L}} [Ww_a D_a(\psi) + Kk_a S_a(\psi)] \tag{5}$$

Subject to $\psi(c,\theta,\phi)$

$$\in \Omega_0; \begin{cases} c_{\min} \leq c \leq c_{\max} & \text{cycle time constraints} \\ 0 \leq \theta \leq c & \text{values of offset constraints} \\ \phi_{\min} \leq \phi \leq \phi_{\max} & \text{green time constraints} \\ \sum_{i=1}^{m} (\phi + I)_i = c & \forall\, m = \mathbf{M},\ \forall\, n \in \mathbf{N} \end{cases}$$

Variable Discretization

Common network cycle time

$$c = c_{\min} + \Phi_i(c_{\max} - c_{\min})/2^{l_i} - 1 \quad i = 1 \tag{6}$$

For offsets

$$\theta = \Phi_i c(2^{l_i} - 1) \quad i = 2,3,\ldots,N_j \tag{7}$$

Mapping the vector of offset values to a corresponding signal stage change time at every junction is carried out as follows:

$$\theta_i = S_{i,j}; \quad i = 1,2,\ldots,N_j, \quad j = 1,2,\ldots,m \tag{8}$$

where $S_{i,j}$= signal stage change time at every junction.

For stage green timings

Let $p_1, p_2, \ldots, p_i$ be the numbers representing by the genetic strings for m stages of a particular junction, and $I_1, I_2, \ldots, I_m$ be the length of the intergreen times between the stages.

The binary bit strings (i.e., $p_1, p_2, \ldots, p_i$) can be encoded as follows first;

$$p_i = c_{\min} + \Phi_i(c_{\max} - c_{\min})/(2^{l_i} - 1), \quad i = 1,2,\ldots,m$$

Then, using the following relation, the green timings can be distributed to all the signal stages in a road network as follows second:

$$\phi_i = \phi_{\min,i} + \left(p_i \bigg/ \sum_{i=1}^{m} p_i\right)\left(c - \sum_{k=1}^{m} I_k - \sum_{k=1}^{m} \phi_{\min,k}\right) \qquad i = 1,2,\ldots,m \tag{9}$$

Translation of Signal Timings into TRANSYT Variables

A decoded genetic string is required to translate into the form of TRANSYT inputs, where TRANSYT model accepts the green times as stage start times, hence offsets between signal-controlled junctions. The assignment of the decoded genetic strings to the signal timings is carried out using the following relations in the GATHIC.

1. For road network common cycle time

   $$c \leftarrow u(i,j) \quad i = 1, \quad j = 1,2,3,\ldots,pz$$

   where u represents the corresponding decoded parent chromosome, j represents the population index, and i represents the first individual in the chromosome set.
2. For offset variables

   $$\theta_n(i,j) \leftarrow u(i,j); \quad i = 2,3,4,\ldots,N_j, \quad j = 1,2,3,\ldots,pz$$

   Since there is no closed-form mapping for offset variables, it is common to map these values to the interval $(0,c)$, hence offset values are mapped using Eq. (8). The decoded offset values are in some cases higher than the network cycle time due to the coding process in the GA. In this case, the remainder of a division between $u(i,j)$ and the c is assigned as a stage change time as:

   $$\theta_n(i,j) \leftarrow \text{MOD}[u(i,j),c]; \quad i = 2,3,4,\ldots,N_j, \quad j = 1,2,3,\ldots,pz$$
3. For green timing distribution to signal stages as a stage change time is

   $$\theta_{n,m}(i,j) = \theta_{n,m-1}(i,j) + ((I+\phi)_{n,m}(i,j)) \leq c; \quad \forall\, n \in \mathbf{N},\ \forall\, m \in \mathbf{M},\ i = 1,2,3,\ldots,m$$

Optimization Steps

The steps of GATHIC are set out below:

- Step 0. Initialization. Define the permissible range ($\psi_{\min}$ to $\psi_{\max}$) for the decision variables.

There are no clear theoretical formulae for the appropriate population sizing, but Carroll's (1996) suggestion for this kind of problems is between 10 and 50.

- Step 1. Generate the initial random population of signal timings $\mathbf{X}_t$; set t=1.

All binary bits for each chromosome are initialized randomly

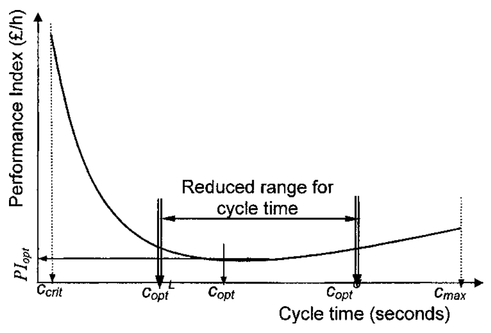


Fig. 1. Common cycle time versus network performance index

using a random number generator. Due to the given minimum and maximum bounds for the signal timing variables as an input to the GATHIC, the generated sequence for signal timings are not likely to produce infeasible sets. If signal-timing constraints do not satisfy for generated signal settings, the GATHIC will automatically discards those generated signal timings by way of TRANSYT program.

- Step 2. Decode all signal timing parameters using Eqs. (6), (7), and (9) to map the chromosomes to the corresponding real numbers.
- Step 3. Run TRANSYT.
- Step 4. Get the network PI.
- Step 5. Calculate the F for each chromosome x_j using Eq. (4).
- Step 6. Reproduce the population in proportion to the F values.
- Step 7. Carry out the crossover operator by a random choice with probability p_c.

Crossover probability (denoted by p_c) is defined as the ratio of the number of offspring produced in each generation to the population size. This ratio controls the expected number $p_c^* pz$ of chromosomes to undergo the crossover operation. A higher crossover rate allows exploration of more of the solution space and reduces the chances of settling for a bad local optimum, but the higher the crossover rate, the longer the computation time. Based on previous studies, Goldberg (1989) and Carroll (1996) set the probability of crossover (p_c) between 0.5 and 0.8. Hence, p_c is selected as 0.5 in this study.

- Step 8. Carry out the mutation operator by a random choice with probability p_m.

Mutation probability (denoted by p_m) is a parameter that controls the probability with which a given string position alters its value. If p_m is too low, many genes that would have been useful are never tried out; but if it is too high, there will be much random perturbation, the offspring will start losing their resemblance to the parents, and the algorithm will lose the ability to learn from the history of the search. p_m can be set to $1/pz$ (Carroll 1996).

- Step 9. Carry out the *elitism* operator if the best fit individual has replicated; if not, a random individual is chosen and the chromosome set of best parent is mapped into that individual.
- Step 10. If the difference between the population average fitness and population best fitness index is less than 5% then go to the Step 11, otherwise go to Step 2.
- Step 11. If the maximal generation number is achieved or $|\max F - \text{Average } F| \leq 0.0001$, then the chromosome with the highest fitness is adopted as the optimal solution of the problem. Or else, increase the generation number by one and return to Step 2.

The main disadvantage of the GATHIC model is the CPU time which increases when it is combined with the HC algorithm. One of the main reasons for this is that the search space for the GATHIC is quite large when it is set in the usual way (Ceylan and Bell 2004a,b). Therefore, if the search space is decreased analytically, the CPU time for the GATHIC may considerably be decreased. Thus, the ADESS is developed. It seeks the lower and upper bounds for the signal timing variables by way of performance index and common cycle length.

A typical cycle length and the performance index=delay curve can be seen in Fig. 1. For consistency with the traffic model of TRANSYT, let $c_{\max}$ be the longest acceptable common cycle length, and for the junction being considered let c_{opt} be the TRANSYT optimal cycle length and c_{crit} be the critical cycle length. If $c_{\text{opt}} < c_{\max}$, the network PI follows the right-hand side of Fig. 1, and the PI becomes very large when the cycle time approaches c_{crit}. The PI attains minimum delay, PI_{opt}, when the cycle time is equal to c_{opt}.

Let c_{opt}^L and c_{opt}^U be the lower and upper bound for the *reduced range* for the GATHIC model parameter space. To obtain the reduced range, the ADESS algorithm is given as

- Step 1: Set $c = c_{\min}$ and $\Delta c = 1$;
- Step 2: Read signals, flows and turning flows data;
- Step 3: Run TRANSYT;
- Step 4: Get the network PI; and draw it versus cycle length curve as in Fig. 1;
- Step 5: If $c = c_{\max}$; then stop; else $c = c_{\min} + \Delta c$ and go to Step 1;
- Step 6: Find the minimum and maximum c_{opt} values, and adjust the GATHIC parameter space as $c_{\min} = c_{\text{opt}}^L$ and $c_{\max} = c_{\text{opt}}^U$; and
- Step 7: Follow the previously given GATHIC steps (see Fig. 2).

At Step 1, c_{crit} is given as $c_{\min}$ for this study.

The flowchart of the ADESS algorithm is given in Fig. 2. The ADESS algorithm performs the process until the prespecified number of iterations are completed. During the run of the algorithm, the signal settings constraints should be satisfied due to practicability and safety reasons. The ADESS finds the minimum PI versus c_{opt}, which is the TRANSYT optimal cycle length, then the parameter ranges for the GATHIC is reduced to between c_{opt}^L and c_{opt}^U by taking the 15th value before c_{opt} and 25th value after c_{opt}. The reason for taking the 15th and 25th values is that GATHIC optimal cycle length lies between those ranges, which are empirically determined; therefore, it would be enough to reduce the search space to those values.

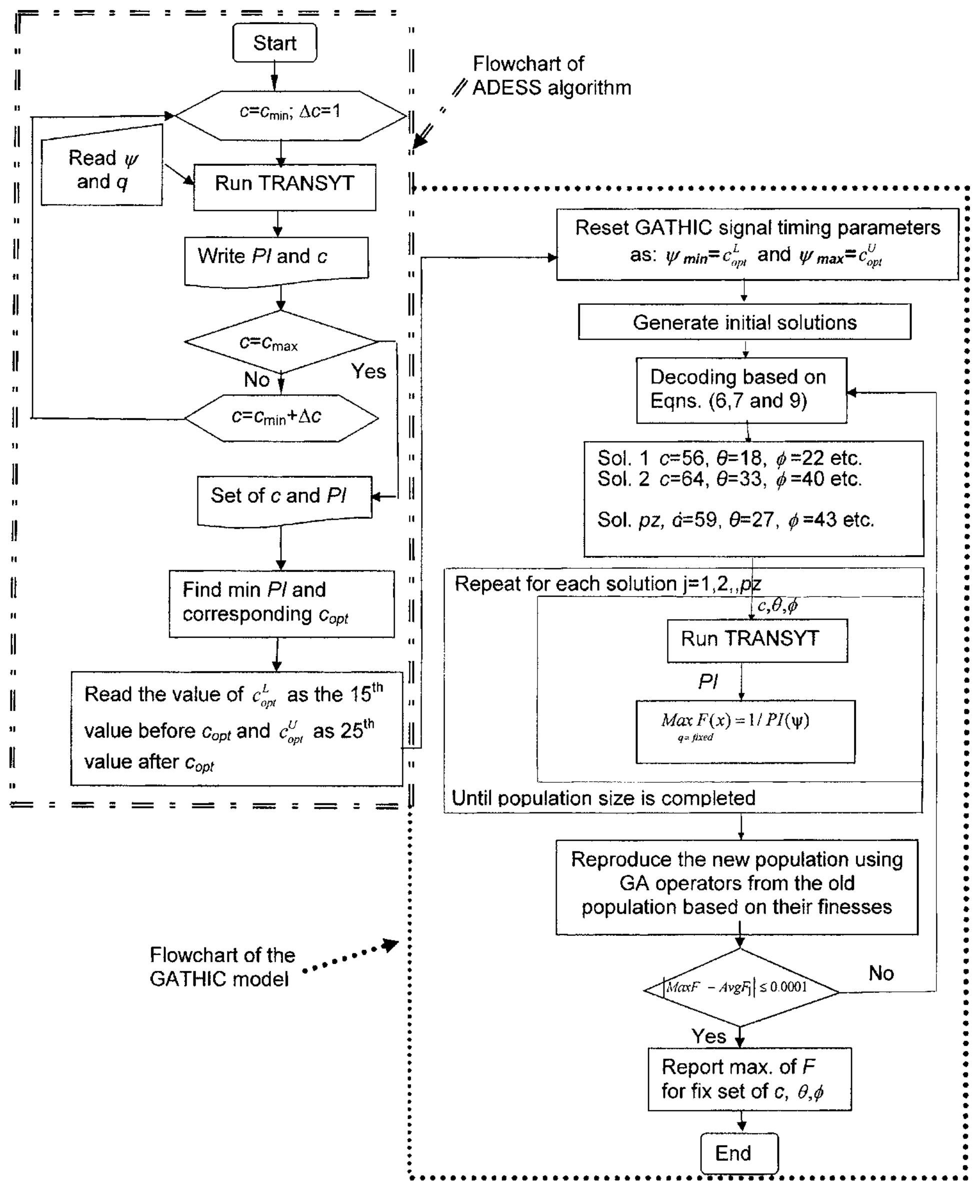
Numerical Application

A test network is chosen based upon the one used by Allsop and Charlesworth (1977). Basic layouts of the network and stage configurations are given in Figs. 3 and 4. A set of fixed link flows are given in Table 1. This numerical test includes 21 signal setting variables at 6 signal-controlled junctions. The GATHIC model performance is also tested for decreased and increased values of Table 1 by 10%.

GATHIC Application without ADESS Algorithm

The GATHIC parameter ranges are given as:

$c_{\min}, c_{\max} = 36, 135$ s	Common network cycle time
$\theta_{\min}, \theta_{\max} = 0, 135$ s	Offset values
$\phi_{\min} = 7$ s	Minimum green time for signal stages
$I_i = 5$ s	Intergreen time between the stages



Start
$c=c_{min}$; $\Delta c=1$
Read ψ and q
Run TRANSYT
Write PI and c
$c=c_{max}$
No
Yes
$c=c_{min}+\Delta c$
Set of c and PI
Find min PI and corresponding c_{opt}
Read the value of c^L_{opt} as the 15th value before c_{opt} and c^U_{opt} as 25th value after c_{opt}
Flowchart of ADESS algorithm
Reset GATHIC signal timing parameters as: $\psi_{min}=c^L_{opt}$ and $\psi_{max}=c^U_{opt}$
Generate initial solutions
Decoding based on Eqns. (6,7 and 9)
Sol. 1 c=56, θ=18, ϕ=22 etc.
Sol. 2 c=64, θ=33, ϕ=40 etc.
Sol. pz, c=59, θ=27, ϕ=43 etc.
Repeat for each solution j=1,2,,pz
c,θ,ϕ
Run TRANSYT
PI
$Max\,F(x)=1/PI(\psi)$ $q=fixed$
Until population size is completed
Reproduce the new population using GA operators from the old population based on their finesses
$|MaxF - AvgF_i| \leq 0.0001$
No
Yes
Report max. of F for fix set of c, θ,ϕ
End
Flowchart of the GATHIC model

Fig. 2. Flowchart of the ADESS algorithm and the GATHIC model

Based on previous studies (Goldberg 1989; Carroll 1996), the GATHIC is performed with the following GA parameters in all cases:

Population size (pz)	=	20;
Reproduction operator	=	binary tournament selection;
Crossover operator	=	uniform crossover;
Probability of crossover (p_c)	=	0.5;
Probability of mutation (p_m)	=	$1/pz$=0.05;
Bits per timing parameter	=	8;
Number of timing variables	=	21;
Chromosome length	=	168 bit;
The maximal number of generation (t)	=	40.

Eight-bit representation of timing parameters are chosen in this study. The reason for choosing the eight-bit representation of the parameters is to increase the precision per design parameter.

The application of the GATHIC to the example network can be seen in Fig. 5, where the convergence of the algorithm and improvement on the network performance index and hence the signal timings can be seen. Model analysis is carried out for the 40th generation, and network performance index obtained for that generation is 672.0 £/h. The shuffling process began after the 13th generation and there was not much improvement to the population best fitness previously found. The reason for this is that in the first iterations, the algorithm finds a chromosome with good fitness value which is better than average fitness of the population. The algorithm keeps the best fitness then starts to improve population average fitness to the best chromosome while improving the best chromosome to optimum or near optimum. The considerable improvement on the objective function usually takes place in the first few iterations because the GATHIC start with randomly generated chromosomes in a given population pool. After that, small improvements to the objective function takes place since the average fitness of the whole population will push forward the population best fitness by way of genetic operators, such as mutation and crossover.

Table 2 shows the signal timings and the final value of the performance index in terms of £/h and veh-h/h. The common

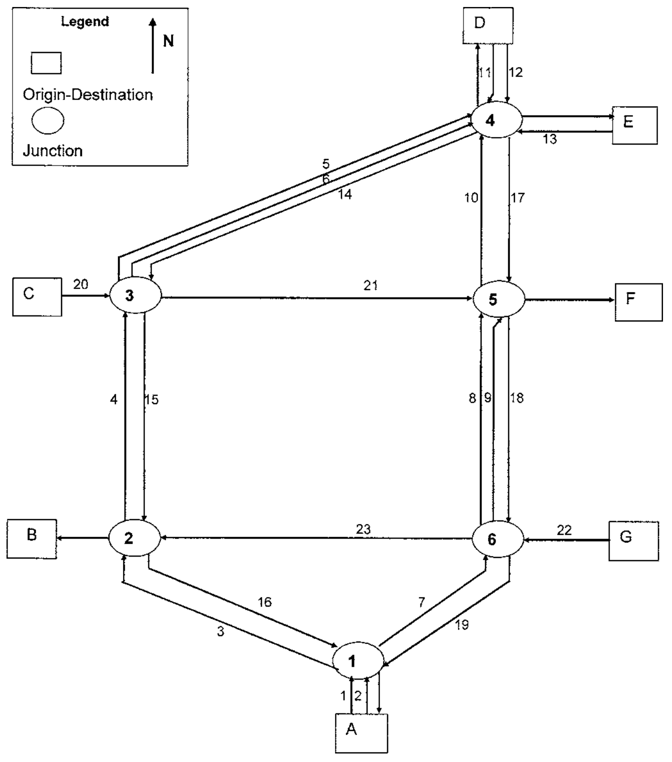


Fig. 3. Layout for the test network

network cycle time obtained from the GATHIC application is 56 s and the start of greens for every stage in the signalized junction is presented in Table 2.

As for the computation efforts, the GATHIC performed on P4 2800 MHz personal computer in FORTRAN 90. The total computation effort for complete run of the GATHIC model run was 33.6 min without revised search space.

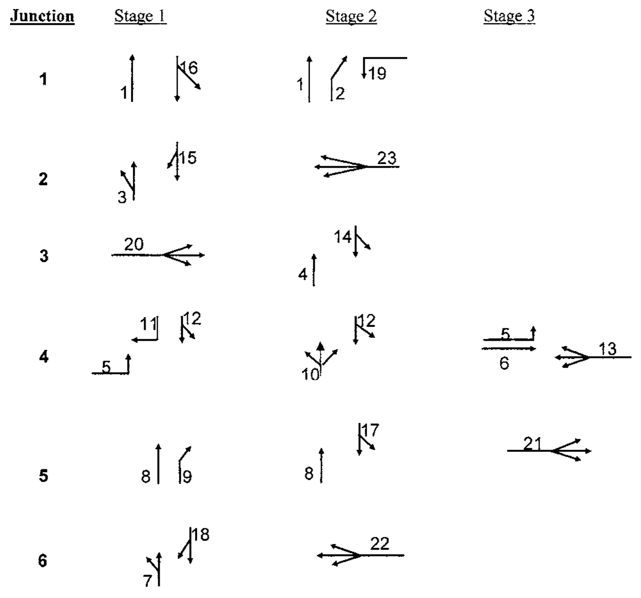


Fig. 4. Stage configurations for the test network

Table 1. Fixed Set of Link Flows

Link Number	Flow (veh/h)	Saturation flow (veh/h)	Free-flow travel time c^0 (s)
1	716	2,000	0[a]
2	463	1,600	0[a]
3	716	3,200	10
4	580	3,200	15
5	636	1,800	20
6	174	1,850	20
7	463	1,800	10
8	478	1,850	15
9	121	1,700	15
10	478	2,200	10
11	500	2,000	0[a]
12	249	1,800	0[a]
13	450	2,200	0[a]
14	791	3,200	20
15	793	2,600	15
16	669	2,900	10
17	407	1,700	10
18	346	1,700	15
19	619	1,500	10
20	1,290	2,800	0[a]
21	1,056	3,200	15
22	1,250	3,600	0[a]
23	840	3,200	15

[a]These are the entry links and no travel time are given during the calculations in TRANSYT.

GATHIC Application with ADESS Algorithm

The application of the ADESS algorithm in an example road network provided the possible ranges of c_{opt}^{L} and c_{opt}^{U} that may lead to narrowing the search space for the GATHIC in order to locate good local optimum with less CPU time. The output of the ADESS is further processed for subsequent use in the GATHIC as

$$c = c_{\text{opt}}^{L} + \Phi_i \frac{(c_{\text{opt}}^{U} - c_{\text{opt}}^{L})}{2^{l_i} - 1} \qquad i = 1$$

and the reduced lower and upper bounds for signal timings are

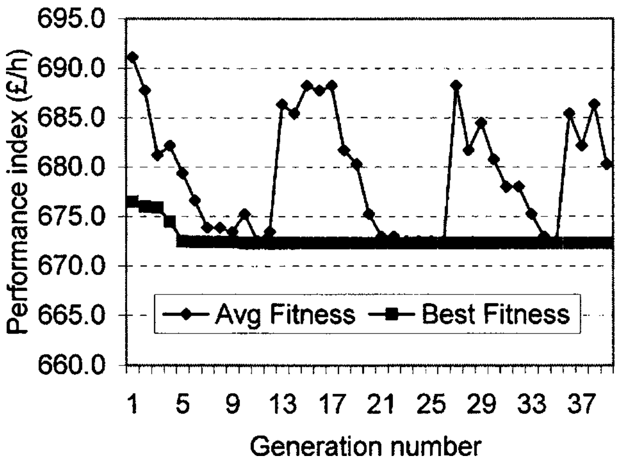


Fig. 5. Convergence of the GATHIC

Table 2. Final Values of Signal Timings Derived from the GATHIC

Performance index		Cycle time	Junction number	Start of green (s)		
				Stage 1	Stage 2	Stage 3
£/h	veh-h/h	c (s)	r/i	$S_{i,1}$	$S_{i,2}$	$S_{i,3}$
672.0	69.3	56	1	21	43	—
			2	9	39	—
			3	18	52	—
			4	27	48	10
			5	22	33	53
			6	51	21	—

$c^L_{\text{opt}}, c^U_{\text{opt}} = 54,84$ s Common network cycle time

$\theta_{\min}, \theta_{\max} = 0,84$ s offset values

$\phi_{\min} = 7$ s

$I_i = 5$ s $i = 1,2,3,\ldots m$

The application of the GATHIC in this section is carried out by running the ADESS and GATHIC models. The network performance index and signal timings are given in Table 3. The network performance index is improved to a value of 666.6 £/h and the cycle length is 56 s. The CPU time is reduced to the 13.8 min. The CPU improvement over the unrevised GA search space is about 60% and the performance improvement is about 1%.

The results of the 10% increased and decreased values of demand from Table 1, by way of GATHIC with ADESS algorithm, are given in Table 4. The performance index is improved when the demand in Table 1 is decreased by 10% with a decreased CPU and cycle length. The CPU increases when the demand increases. This happens due to the TRANSYT program that evaluates the network PI in a longer time than the lower demand from the base value.

Conclusions

This study solves the area traffic control problem by combining GA and HC methods. The GATHIC model is developed and its disadvantage in terms of CPU time may be removed by introducing the ADESS algorithm to reduce search space. The ADESS with GATHIC is an optimization heuristic to optimize the network common cycle length by taking into account coordination both in network and at individual junction level simultaneously. Other lower and upper bounds for signal timing parameter ranges are also possible in GATHIC methodology, but setting the signal timing parameters at the original level considerably increases the CPU time from about 13 min to 37 min for this example. CPU time improvement of the GATHIC is about 60% with ADESS algorithm. The GATHIC provides signal timings for further use in TRANSYT-HC method. The convergence of the model may be guaranteed for this example due to the one-step solution algorithm for signal timing optimization.

Table 3. Final Values of Signal Timings Derived from the GATHIC with ADESS Algorithm

Performance index		Cycle time	Junction number	Start of green (s)		
				Stage 1	Stage 2	Stage 3
£/h	veh-h/h	c (s)	i	$S_{i,1}$	$S_{i,2}$	$S_{i,3}$
			1	3	25	—
			2	47	21	—
666.6	68.3	56	3	55	34	—
			4	9	30	48
			5	4	14	34
			6	33	3	—

Table 4. Performance of the GATHIC with ADESS Algorithm for Variable Demand

10% decreased values of Table 1				10% increased values of Table 1			
Peformance Index		Cycle	CPU	Peformance index		Cycle c	CPU
£/h	veh-h/h	c (s)	(min)	£/h	Veh-h/h	(s)	(min)
531.1	53.8	53	8.25	881.1	93.7	76	19.65

Although TRANSYT-7F introduced the GA optimization method, the GATHIC combines the GA with HC and introduces an algorithm to reduce the search space for the GA in order to cope with the CPU disadvantage during the optimization process. The main advantages of the GATHIC algorithm over TRANSYT is that it produces the initial set of signal timings for HC algorithm, and it also optimizes the network common cycle time, where each change on the cycle time is consequently evaluated by GATHIC with ADESS algorithm.

The HC optimization technique may be more effectively used with a GA notion with the cost of CPU time, but optimal or near-optimal solution of the signal timings parameters may be obtained without using a complex procedure as given in Heydecker (1996). Although the CPU time can be reduced by introducing the ADESS algorithm together with GATHIC, the CPU time is considerably high over traditional methods.

It is also obtained that the GATHIC application for the example network is better to locate an initial starting point for TRANSYT optimization routine. In this way, the deficiency of TRANSYT to initial signal timings can also be relaxed. It is well known that TRANSYT provides local optimum values only and these values depend on the selection of the starting point. Furthermore, there is no information available on the relative error with respect to the global optimum of the solution found.

The GATHIC model, which takes full advantage of flexibility by obtaining each of the optimal or near-optimal signal timings at a signal-controlled road network, is the simultaneous optimization of control variables at the network and individual junction level with considering coordination of closely spaced junctions.

For this small network, the effect of stage ordering is not taken into account due to the small numbers of the stage ordering permutations and the difficulties in GA coding in the developed GATHIC algorithm. Further study should take into account the effect of stage ordering in signal timing optimization using the permutation coding (in this study binary coding is used). Further study should also be on testing GATHIC with realistic size networks.

Acknowledgments

The writer is grateful for anonymous referees for their helpful comments on an earlier draft of this paper. Scientific and Technological Research Council of Turkey (TUBITAK) with Project No.

104I119 and The Scientific Research Unit of Pamukkale University (BAP) are also acknowledged for providing research Grant No. 2003MHF008.

Notation

The following symbols are used in this paper:

c	=	common cycle timer (s);
c_{crit}	=	critical value of common cycle time (s);
$c_{\max}$	=	maximum acceptable common cycle time (s);
$c_{\min}$	=	minimum acceptable common cycle time (s);
c_{opt}	=	optimal cycle time (s);
D_a	=	delay in vehicles-hours/hour on link a in $\mathbf{L}$;
I_m	=	intergreen time (s);
K	=	overall cost per 100 veh-stops;
k_a	=	stop weighting factors on link a per unit time;
$\mathbf{L}=\{1,2,3,\ldots,N_L\}$	=	set of N_L links where each traffic stream approaching any junction is represented by its own link;
l	=	length of chromosome;
$\mathbf{M}$	=	total number of stages at all signal controlled junction;
m	=	number of stages at each signalized junction;
$\mathbf{N}=\{1,2,3,\ldots,N_j\}$	=	set of N_j nodes each of which represents a signal-controlled junction;
PI	=	network performance index (£/h or veh/h);
pz	=	size of population;
$\mathbf{q}$	=	vector of average arrival flow (in vehicles/hour) for link a in $\mathbf{L}$;
S_a	=	number of vehicle stops on link a in $\mathbf{L}$;
T	=	period (h) over which the estimation is made;
W	=	overall cost per average veh h of delay;
w_a	=	delay and stop weighting factors on link a per unit time;
x	=	vector of chromosomes in the population;
z	=	number of control variables in an signalized network;
$\Delta\psi_i$	=	precision of design variable;
$\psi_{i,\min}$	=	whole lower bound vector of signal timings;
$\psi_{i,\max}$	=	whole upper bound vector of signal timings;
$\varphi=[\phi_m;\ \forall n\in\mathbf{N}]$	=	vector of green times, where element $\phi_{i,n}$ is the duration of green for stage i at junction n;
$\lfloor\theta=\theta_{1,n};\ \forall n\in\mathbf{N}\rfloor$	=	vector of start of green for stage 1 at each junction n relative to an arbitrary time origin for the network as a whole, i.e., signal offset;
$\psi=(c,\theta,\varphi)$	=	vector of feasible signal timings;
$\phi_{\max,m}$	=	maximum allowable green time for each stage m (s);
$\phi_{\min,m}$	=	minimum allowable green time for each stage m (s);
Ω_0	=	whole vector of feasible signal timings; and
Θ_i	=	integer number resulting from binary representation of the timing variables.

References

Akcelik, R. (1981). "Traffic signals: Capacity, and timing analysis." *ARR 123*, Australian Road Research Board, Vermonth South, Victoria, Australia.

Allsop, R. E. (1992). "Evolving application of mathematical optimization in design and operation of individual signal-controlled road junctions." *Mathematics in transport planning and control*, J. D. Griffths, ed, Clarendon, Oxford, U.K., 1–25.

Allsop, R. E., and Charlesworth, J. A. (1977). "Traffic in a signal-controlled road network: An example of different signal timings including different routings." *Traffic Eng. Control*, 18(5), 262–264.

Carroll, D. L. (1996). "Genetic algorithms and optimizing chemical oxygen-iodine lasers." *Developments in theoretical and applied mechanics*, Vol. XVIII, H. B. Wilson, R. C. Batra, C. W. Bert, A. M. J. Davis, R. A. Schapery, D. S. Stewart, and F. F. Swinson, eds., School of Engineering, The Univ. of Alabama, Tuscaloosa, 411–424.

Ceylan, H., and Bell, M. G. H. (2004a). "Traffic signal timing optimization based on genetic algorithm approach, including drivers' routing." *Transp. Res., Part B: Methodol.*, 38(4), 329–342.

Ceylan, H., and Bell, M. G. H. (2004b). "Reserve capacity for a road network under optimized fixed time traffic signal control." *J. Intell. Transportation Syst.*, 8(2), 87–99.

Gallivan, S., and Heydecker, B. G. (1988). "Optimizing the control performance of traffic signals at a single junction." *Transp. Res., Part B: Methodol.*, 22(5), 357–370.

Gen, M., and Cheng, R. (1997). *Genetic algorithms and engineering design*, Wiley, New York.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimisation, and machine learning*, Addison-Wesley, U.K.

Goldberg, D. E., and Deb, K. (1991). "A comparative analysis of selection schemes used in genetic algorithms." *Foundations of genetic algorithms*, G. J. E. Rawlins, ed., Morgan Kaufmann, San Mateo, Calif., 69–93.

Heydecker, B. G. (1996). "A decomposed approach for signal optimisation in road networks." *Transp. Res., Part B: Methodol.*, 30(2), 99–114.

Heydecker, B. G. (1992). "Sequencing of traffic signals." *Mathematics in transport planning and control*, J. D. Griffths, ed., Clarendon, Oxford, U.K., 57–67.

Heydecker, B. G., and Dudgeon, I. W. (1987). "Calculation of signal settings to minimize delay at a junction." *Proc., 10th Int. Symp. On Transportation and Traffic Theory*, Elsevier, Amsterdam, 159–178.

Robertson, D. I. (1969). "TRANSYT: A traffic network study tool." *RRL Rep., LR 253*, Transport and Road Research Laboratory, Crowthorne, U.K.

Silcock, J. P., and Sang, A. P. (1990). "SIGSIGN: A phase-based optimization program for individual signal-controlled junctions." *Traffic Eng. Control*, 31(5), 291–298.

Vincent, R. A., Mitchell, A. I., and Robertson, D. I. (1980). "User guide to TRANSYT version 8." *TRRL Rep., LR888*, Transport and Road Research Laboratory, Crowthorne, England.

Wong, S. C. (1996). "Group-based optimisation of signal timings using the TRANSYT traffic model." *Transp. Res., Part B: Methodol.*, 30(3), 217–244.