Hybrid Genetic Algorithm To Optimize Signal Phasing and Timing

MOHAMMED A. HADI AND CHARLES E. WALLACE

Signal timing optimization involves the selection of four basic design elements: phase sequence, cycle length, green split, and offset. None of the available signal timing models is considered adequate to optimize all four design elements, particularly in two-dimensional networks. Among the current models, TRANSYT-7F is most effective for timing, but it does not optimize phasing. Researchers have considered several methods for enhancing TRANSYT-7F to include phasing optimization but thus far no method has proven practical. An exhaustive search of possible phasing combinations is computationally prohibitive; thus a new approach is needed. Genetic algorithms (GAs) are heuristic probalistic search procedures that have been applied to a wide range of engineering problems. The use is investigated of a GA in combination with the TRANSYT-7F optimization routine to select all signal timing design elements. The main purpose of the GA in the proposed scheme is to optimize phase sequences. Two implementations of the GA model are presented. In the first, the GA and TRANSYT-7F optimization routines are executed concurrently to achieve an optimal solution. In the second, the GA is allowed to optimize cycle length, phase sequences, and offsets. Then TRANSYT-7F is used to adjust the resultant signal timing. The results suggest that both implementations have potential for optimizing signal phasing and timing. However, the first method produces more consistent results. It also requires longer execution time.

Several computer models have been developed for off-line optimization of signal timing. Each of these models has its own area of application and its own particular strengths and weaknesses. For coordinated systems, the optimization models currently used include PASSER II (*1,2*), MAXBAND (*3*), and TRANSYT-7F (*4*). None of these models is all inclusive, and a combination of them is often used to achieve a desired design policy (*5–9*).

PASSER II and MAXBAND select a signal timing plan by maximizing bandwidth efficiency (the ratio of total bandwidth to the cycle length) and have been used primarily for arterial streets. Although MAXBAND has been extended for application to multiple-arterial networks (*10*), this version is not widely used, primarily because of excessive computer time and its current limitation to mainframe computers.

Both MAXBAND and PASSER II can optimize cycle length, phase sequences, and offsets. The main advantage of these programs is their ability to optimize phase sequences. One important disadvantage is that maximizing bandwidth does not necessarily result in optimal system performance in terms of stops, delay, and fuel consumption. This is because the maximal bandwidth design strategy does not explicitly recognize traffic demand as a function of time on individual links. In addition, these programs do not optimize green splits, although they do calculate splits based on manipulating the degrees of saturation.

TRANSYT-7F has been used for arterial streets as well as two-dimensional networks. Traditionally, TRANSYT-7F optimizes cycle length, splits, and offsets by minimizing a disutility index (DI), which is a function of delay, stops, fuel consumption, and, optionally, queue spillover. In the latest release of the program, a progression-based optimization model that uses the progression opportunities (PROS) concept has been implemented, overcoming an earlier shortcoming of the model (*11,12*). This concept expands upon the maximal bandwidth approach by considering short-term progression opportunities within the system.

In spite of this major enhancement, the most significant deficiency in TRANSYT-7F remains—its inability to optimize phase sequences. Phasing is an input to the model and cannot be changed in a given run. To examine alternative sequences, they have to be explicitly coded in multiple runs. Cohen (*9*) noted that adding a phase sequence optimization capability to TRANSYT-7F would involve combining a linear gradient search technique with a combinatorial problem. He recognized that this appears to be computationally infeasible since there are 4^n possible phase sequence combinations at n intersections, assuming four possible phase sequences, namely, leading left, lagging left, leading in one direction and lagging in the other, and vice versa.

To deal with this problem, other computer programs such as PASSER II and MAXBAND have been used as a "preprocessor" to determine the phase sequence before TRANSYT-7F is run to optimize signal timing. It has been shown that this method has the potential for improving the signal timing plan produced by TRANSYT-7F (*8,9*). This strategy has been particularly successful for designing arterial signal timing.

For this purpose, model integration programs like the Arterial Analysis Package (AAP) have been used to run PASSER II and TRANSYT-7F from a common data base, thus reducing the effort required to implement this strategy (*13*). PASSER II can be run from the AAP, which can import the phasing and timing; then these can be automatically mapped as the initial timing for a TRANSYT-7F optimization run. This also has the advantage of providing TRANSYT-7F with a better "starting point," which has also been shown to lead to potentially better solutions (*5,6*), although with the new PROS options this factor is less important (*11*).

For networks, the development of integration models similar to the AAP has been delayed because of the unavailability of a phase sequence optimization program that can deal with networks and run efficiently on microcomputers.

Transportation Research Center, University of Florida, 512 Weil Hall, Gainesville, Fla. 32611.

(Note that FHWA will be developing such a package in the near future.)

The dilemma facing software developers is how to overcome these functional gaps, particularly for networks:

- Merging the functions of PASSER II or its network counterpart, PASSER IV, which is under development [reported functionally by Chaudhary et al. elsewhere (*14*)], with TRANSYT-7F. This does not seem to be practical for both functional and programmatic reasons, not to mention the problem of software "ownership."
- Improving upon the integrated application of a bandwidth model (PASSER IV or MAXBAND) and TRANSYT-7F to be more automatic. This is certainly feasible, but it will still require user intervention.
- Putting phase sequence optimization explicitly into TRANSYT-7F. This is technically possible, but the massive restructuring of the model makes the chances for a timely and successful undertaking unlikely.
- Applying a new approach to using TRANSYT-7F in an iterative fashion to optimize phasing by trial and error. As mentioned earlier, the possible combinations seem to make this approach prohibitive unless a more effective method than exhaustive search can be found.

The last possibility is the aim of this paper. A new approach is applied to an iterative, heuristic algorithm for phase sequence optimization using TRANSYT-7F.

BACKGROUND OF PROPOSED METHOD

Heuristic search strategies called genetic algorithms (GAs) are finding increasing application in a variety of problems in science, engineering, business, and the social sciences (*15*). The GA strategies are based on the mechanics of natural selection and natural genetics. A number of analytical and empirical studies have demonstrated their capability in function optimization and artificial intelligence applications.

The number of solutions evaluated using GAs to locate a satisfactory solution to a given problem is small in comparison with the size of the search space. Nevertheless, the computational requirements for GAs can be severe. A hybrid scheme that switches from the genetic search to a conventional nonlinear programming approach has been suggested in the literature to deal with this problem (*15*).

Foy et al. (*16*) investigated the implementation of a GA to produce optimal, or near optimal, signal timing. In that study, the GA was used to optimize cycle length, splits, and course offsets (actually simply flipping the two phases with no explicit offsets) for two-phase operations for a four-intersection network. In the optimization process, a simple microscopic simulation model was used to evaluate alternative solutions based on minimizing delay.

The results of Foy et al. show an improvement in the system performance when this GA strategy is used and suggest that GAs have potential in optimizing signal timing. The results obtained, however, were not compared with those that could be achieved using existing optimization models. In addition, the GA model was applied to a very simple system with two-phase operation and no explicit offsets between intersections. More typical real-world applications of the GA model are needed to prove its effectiveness.

In this study, a GA was used in conjunction with the TRANSYT-7F hill-climbing routine to optimize all four signal timing parameters (offset, split, cycle length, and phase sequence). This hybrid scheme works with arteries as well as two-dimensional networks and can optimize timing based on system performance (TRANSYT-7F DI), PROS, bandwidth, or a combination of these.

TRANSYT-7F MODELS

TRANSYT-7F is one of the most powerful computer programs for traffic signal timing and traffic flow analysis. Traditionally, TRANSYT has consisted of two main parts: a traffic flow model and an optimization model.

The traffic flow model in TRANSYT-7F is a deterministic, macroscopic, time-scan simulation. It simulates traffic flow in a street network of signalized intersections to compute a DI for a given signal timing and phasing plan. This DI is a function of stops, delay, fuel consumption, and, optionally, queue spillover.

The TRANSYT-7F optimization procedure is based on an iterative, gradient search technique known as hill-climbing. It makes changes to the signal timing to determine whether a performance index (PI) has improved. By adopting only those changes that improve the PI, the optimizer tries to find a set of timing that optimizes the PI subject to any constraints.

In the latest release of the program (*4*), an improvement was added that allows it to calculate the PROS value for multiarterial networks. PROS is a measure of progression that considers not only through bands but also short-term progression opportunities within the system (*11,12,17*).

In the past, the PI used in the optimization routine of TRANSYT-7F was always the DI calculated by the traffic flow model. However, the latest release of the program has the ability to optimize signal timing on the basis of PROS only, PROS and DI, PROS or DI, and DI only. It has been shown that the PROS-related strategies produce significant improvements in progression compared with DI-only optimizations for single arteries as well as for multiarterial networks (*11*).

ELEMENTS OF GENETIC ALGORITHMS

GAs are a family of adaptive search procedures that are loosely based on models of genetic changes in a population of individuals. 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 (or data base) of solutions from which better solutions are created rather than making incremental changes to a single solution to the problem. Thus, they are less susceptible to the local optimum problem experienced by traditional nonlinear programming methods, in particular hill-climbing algorithms like those of TRANSYT. In addition, many optimization techniques are calculus-based and depend on the restrictive re-

quirements of continuity and function derivatives. GAs use only objective function information; thus, they do not need derivatives of the objective function and can work with noisy and discontinuous functions.

GAs are iterative procedures that search by allowing a population (data base) of alternative solutions to reproduce and cross among themselves with biases allocated to the most fit members of the population. Possible solutions within each population are coded as binary strings (chromosomes). For example, if a solution to a certain problem takes a maximum value of, say, 31, it can be coded as a five-bit string (e.g., 11011 for 27).

A solution to a signal optimization problem consists of a cycle length, offsets, phase sequences, and phase splits; thus it requires a multiparameter solution. To code a multiparameter solution as a binary string, each parameter (cycle length, offset, etc.) is first coded as a binary variable as described above. Then these single parameters are concatenated to obtain the required binary string. For example, if six-bit, three-bit, and six-bit binary variables are selected to represent cycle length, phase sequence, and offset, respectively, the binary string representing the solution would be as follows (assuming that all bits in the solution are zeros):

000		000	000000		000000	000000
*n*th	. . .	1st	*m*th	. . .	1st	Cycle
sequence		sequence	offset		offset	length

where m and n are the number of offsets and sequences to be optimized, respectively.

A set of solutions is selected at random in the initial population using "successive coin flips" (heads = 1, tails = 0). Thereafter, during each iteration step, called a generation, the solutions of the current population are evaluated using an objective function, and on the basis of this evaluation, a new population of candidate solutions is formed (see Figure 1). The objective function used in the process is model specific. For example, in signal timing optimization this could be fuel consumption, bandwidth, PROS, or a combination of the foregoing.

The result is a new set of solutions ("offspring"), and the new solutions are more fit (that is, have a better objective function value) than the parent solutions from the previous generation. The GA continues to generate successive populations until some solution criterion is met. This could be reaching a fixed maximum number of generations or being unable to achieve further improvement in the objective function.

Three basic components are required for GA implementation:

1. A scheme to allow for a binary string (a string of 0's and 1's) representation of alternative solutions to the problem;
2. An objective function to evaluate the fitness of each solution; and
3. Genetic operators that mimic the biological evolution process (these operators are used in the formation of successive populations).

A simple GA that produces good results in many problems consists of three genetic operators: reproduction, crossover, and mutation. Further improvements to GAs have been suggested in the literature and more advanced operators and techniques have been implemented (*15*). These include dominance, inversion, intrachromosomal duplication, deletion, translocation, segregation, niche exploitation and speciation, migration, marriage restriction, and sharing functions.

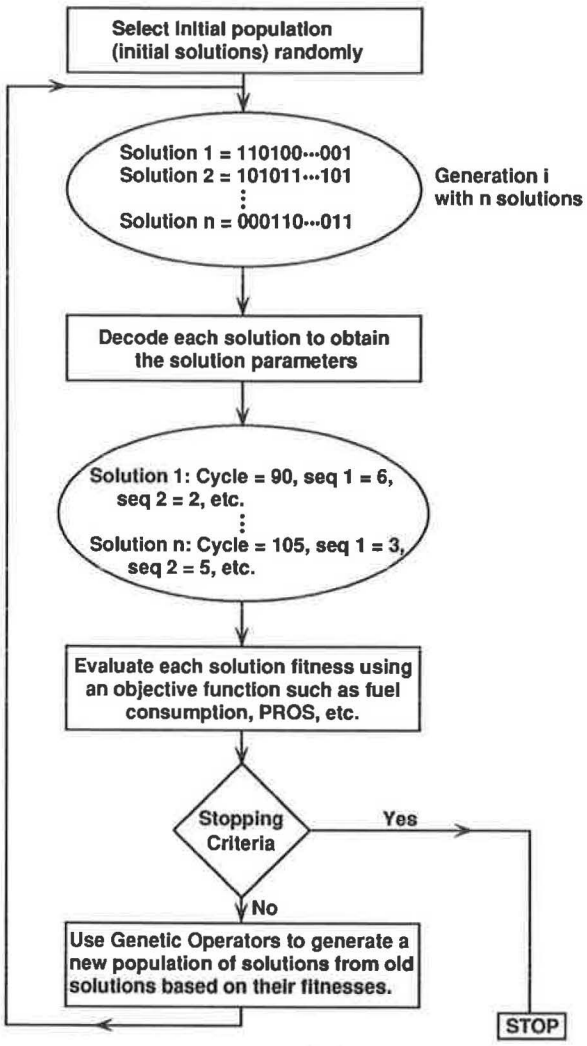


FIGURE 1 Diagram explaining the possible use of a GA in signal timing optimization.

A simple GA that uses the three basic operators (reproduction, crossover, and mutation) is employed in this study. The following is a brief description of the GA operators and techniques used.

Reproduction

Reproduction is the process of selecting those solutions (individual binary strings) within a given generation that will be allowed to contribute to the next generation. The procedure randomly selects these individuals on the basis of their objective function values (fitness). Strings with higher fitness have a higher probability of contributing one or more offspring in the next generation. Thus, the effect of reproduction is to bias the population to contain more-fit members. The

reproduction process is repeated until the number of the selected strings equals the specified population size. The population size is an input to the model.

Crossover

Crossover randomly chooses two individuals from the population selected using the reproduction operator explained in the previous section. These two individuals (parents) are then "mated" to produce two offspring. This is done by choosing a random number K between 1 and the string length (L) less 1 as a "crossover point." The first K bits of Parent 1 and the last L-K bits of Parent 2 are joined to form one offspring. Similarly, the first K bits of Parent 2 and the last L-K bits of Parent 1 are joined to form the other offspring. This procedure is repeated until there is a new population that has the same number of individuals as the old population (the population size).

For example, assume that the string length in a given problem is 8 and the two selected parents are

Parent 1 = 01101011
Parent 2 = 11010100

If the random number selected between 1 and L-1 is 3, then the resulting crossover yields the following offspring, as illustrated by the underlined substrings above for Child 1 and not underlined for Child 2:

Child 1 = 01110100
Child 2 = 11001011

Crossover probability (CP) is a GA parameter that controls the frequency with which the crossover operator is applied. In each new population $CP \cdot N$ individuals undergo crossover, where N is the population size. If CP is too high, good solutions are discarded faster than selection can produce improvements. If the crossover rate is too low, the search may stagnate (*18*).

Mutation

This operator, with a probability equal to the mutation probability (MP), randomly alters the value of a string position (0 to 1 and vice versa). The function of this operator is to prevent any given bit position from remaining forever fixed to a single value over the entire set of solution alternatives.

MP is a parameter that controls the probability with which a given string position alters its value. Small values of MP are always used because a high level of mutation results in an essentially random search (*18*).

Scaling

At the start of a GA exercise, it is common to have a few solutions with much higher fitness than other solutions. This causes the reproduction process to produce a population with a high proportion of the high-fitness individuals, causing premature convergence.

Late in the exercise, a different problem arises. At this near-optimization stage, the population average fitness may become close to the maximum fitness, although there may still be significant diversity within the population. This leads the reproduction process to select the same number of average and best individuals, leading the process to a random walk.

To deal with these two problems, scaling of the objective function has been suggested to keep appropriate levels of competition throughout the process. Three linear scaling mechanisms have been used. These are linear scaling, sigma truncation, and power law scaling (*15*).

MODEL IMPLEMENTATION

A simple GA similar to the one described in the previous section was used in conjunction with the TRANSYT-7F hill-climbing procedure to optimize signal timing. In this scheme, the main objective of the genetic search was to identify the optimal phase sequences, cycle length, and (possibly) the initial offsets. TRANSYT-7F was then used normally to adjust cycle length within a narrow cycle length range and to optimize splits and offsets.

The objective function used by the GA in selecting the optimal phase sequences was the PROS value calculated by TRANSYT-7F release 7. Other objective functions such as the TRANSYT-7F DI either alone or in combination with PROS could be used instead; however, DI calculation requires executing the TRANSYT-7F simulation model, which takes a much longer time than the PROS calculation.

In this study, a separate computer program was written to exercise the GA and call TRANSYT-7F whenever the hill-climbing procedure, the PROS calculation, or both were needed. Two different implementations of the hybrid GA model were tested in this study, as described below.

Method 1: Concurrent Use of GA and TRANSYT-7F Hill-Climbing

In Method 1, each alternative solution within a GA generation consisted of a phase subsequence for each arterial intersection and a cycle length. Offsets and splits for these solutions were calculated using the TRANSYT-7F hill-climbing procedure and the resulting PROS were used as the solution fitness values. The splits were calculated by the TRANSYT-7F internal initial timing routine based on equalizing the degrees of saturation on the critical conflicting links and were fixed.

The first step in using a GA is to choose a scheme for coding each possible solution as a finite-length binary string. In this implementation, each binary string consisted of a six-digit binary number representing a cycle length and a series of three-digit binary numbers representing arterial phase subsequences on all arterial intersections (defined below). An arterial phase subsequence is defined as the order of all phases serving arterial movements. If all the bits in a coded string were 0's, this string would be as follows:

0 0 0		0 0 0	0 0 0	0 0 0 0 0 0
*n*th	. . .	2nd	1st	Cycle
sequence		sequence	sequence	length

The resultant string was of length $6 + (3 \cdot n)$, where n is the total number of subsequences to be optimized. A fixed number of these strings had to be selected for each GA generation. It has been reported (*15*) that even for very large and complex search spaces, GA can rapidly locate structures with high fitness ratings using a generation population of 50 to 100 alternative solutions (binary strings). The population size (the number of alterative solutions within each generation) used in this study was 50.

The selection of these strings was random in the initial generation. In every generation thereafter, the three basic genetic operators (reproduction, crossover, and mutation) were used to create new strings based on the fitness values. As described above, the fitness values used in this implementation were the PROS values obtained from TRANSYT-7F optimization of offsets for each alternative solution within a GA generation. To perform these runs, a procedure was developed to decode each binary string as a cycle length and phase subsequences, which were then used as inputs to TRANSYT-7F.

The six bits representing cycle length were first converted to an integer (N) between 0 and 63. This number was then mapped into an integer between the minimum and maximum cycle lengths as follows:

$$\text{Cycle} = \text{Min} + N \cdot (\text{Max} - \text{Min})/63 \quad (1)$$

where Min and Max are the minimum and maximum allowable cycle lengths, respectively. This number was then rounded into a multiple of 5 sec.

The three-bit binary numbers were first converted to integers between 0 and 7. Each number was then mapped into a phase subsequence using Table 1. (Note that more exotic phasings such as split phasing were not considered in this study, but can be easily added.)

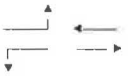
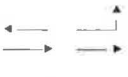
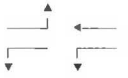
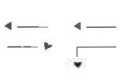
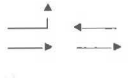
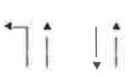
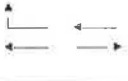
After each TRANSYT-7F run, the GA reads the resultant PROS value from the graphic data file (GDF) produced by TRANSYT-7F. Before these PROS values were used by the GA to evaluate different alternatives, they were scaled using a linear scaling function. This scaling procedure was suggested to improve the GA performance as described above.

In this study, each GA run was executed for a maximum of 50 generations. After the GA had been exercised for the maximum number of generations, the set of phase sequences that produced the highest PROS value was selected as the best phasing.

Method 2: Use of GA Followed by TRANSYT-7F Hill-Climbing

Method 2 differed from Method 1 in that the GA was extended to optimize offsets in addition to cycle length and

TABLE 1 Look-Up Table To Transform Three-Bit Binary Numbers to Phase Subsequences

Binary Number	Integer	Phase Subsequence E-W Artery	Phase Subsequence N-S Artery
000	0		
001	1		
010	2		
011	3		
100	4		
101	5		
110	6		
111	7		

phase sequence. This means that each alternative solution within a GA generation consisted of a cycle length, phase sequences, and offsets.

TRANSYT-7F evaluation runs were performed for each of these solutions to determine their fitness (PROS). The splits in these runs were calculated using the TRANSYT-7F initial timing routine. Offset optimization was performed by the GA, not TRANSYT-7F; thus the computational requirement of this method was much lower than that of Method 1.

In this implementation, each binary string consisted of a six-digit number representing a cycle length, a series of three-digit binary numbers representing phase subsequences, as in Method 1, and a series of six-digit numbers representing offsets for all arterial intersections. For example, if all the bits in a string were 0's, this string would be as follows:

000		000	000000		000000	000000
*n*th	. . .	1st	*m*th	. . .	1st	Cycle
sequence		sequence	offset		offset	length

Each string was of length $(3 \cdot n) + (6 \cdot m) + 6$, where n and m are the number of subsequences and offsets to be optimized, respectively.

The cycle length and phase sequences were decoded from these strings as described in Method 1. To decode offsets, each six-bit binary number representing an offset was first transformed to an integer (M) between 0 and 63. Then it was mapped into an integer between 0 and 100 representing the offset as a percentage of cycle length as follows:

$$\text{Offset} = M \cdot 100/63 \tag{2}$$

This value was then converted into seconds and used together with cycle length and sequences as inputs to TRANSYT-7F.

After the GA had been executed for the maximum number of generations, the four solutions with the highest fitness (PROS) values were located. The TRANSYT-7F hill-climbing routine was then used to adjust the offsets and cycle length within a small cycle-length range. The solutions from these runs were compared and the phase sequences that produced the highest PROS value were selected as the optimal phase sequences. In effect, this method uses the GA to locate the peaks and the TRANSYT-7F hill-climbing procedure to climb them.

Other than the differences mentioned above, all other aspects of Method 2 were the same as those of Method 1.

MODEL APPLICATIONS

Three real-world traffic systems were used to evaluate the two implementations of the hybrid GA discussed in this paper. These systems were Cape Coral Parkway, a seven-intersection artery in the city of Cape Coral, Florida; Volusia Avenue, a 12-intersection east-west artery in the city of Daytona Beach, Florida; and a 12-intersection grid network in the city of Daytona Beach, Florida. For the Daytona Beach network, the objective was to select the phase sequences for two intersecting arteries within the network. These were Ridgewood Avenue, a four-intersection north-south artery, and Volusia Avenue, a three-intersection east-west artery.

In most cases, the existing phase sequences were leading dual lefts without overlap. For the purpose of this study several permitted-only left turns were changed to protected, even though they were not warranted, to provide multiple phasing.

In this comparative study, the splits used were always those calculated by the TRANSYT-7F internal initial timing routine, and they were held constant. For all systems investigated, the cycle range was 100 to 120 sec.

The population size (the number of alternative solutions within each generation), maximum number of generations, crossover probability, and mutation probability used in these studies were 50, 50, 0.90, and 0.01, respectively, based on previous GA research.

The resultant designs from the two GA implementations were compared with those obtained using TRANSYT-7F optimizations with both the "existing" phase sequences and the phase sequences selected by PASSER II-90. In all cases, the objective function used in the optimization was the PROS value calculated by TRANSYT-7F. The comparison was based on perceived progression as measured by the PROS and bandwidth efficiency.

Figure 2 shows the variation of the population maximum fitness and the population average fitness over the generations of evolution when Method 1 was used. For all three systems investigated, this method was able to locate good solutions after only a few GA generations. The solution with the highest fitness could be achieved after 10 to 24 iterations (generations) depending on the system investigated. This means that 500 to 1,200 TRANSYT-7F PROS-only optimization runs were required. This is very efficient compared with the implementation of enumerative schemes in which every possible sequence combination is tried. In that case, the number of possible combinations for a 12-intersection artery and eight possible phase sequences is $(8^{12} = 6.87 \cdot 10^{10})$.

PROS-only optimization is very quick compared with traditional TRANSYT-7F DI optimization. On a 20-MHz 80386 machine with a math coprocessor, it took less than half a second per intersection. For a 12-intersection artery, each PROS optimization took about 6 sec; thus, 1,200 optimization runs would require less than 2 hr. This time can possibly be reduced by further improvement of the GA model as discussed below and by using a quicker TRANSYT-7F optimization that uses fewer optimization steps.

Figure 3 shows the increase in the population maximum PROS and the population average PROS over the GA generations when using Method 2. In this method, after the GA had been run for 50 generations, TRANSYT-7F was used to adjust the resultant timing. This adjustment produced additional improvements in PROS, which can be observed by comparing the maximum PROS values that could be achieved using the GA alone (from Figure 3) with the best PROS values produced by Method 2, as reported in Table 2. These improvements were 13 percent (43 versus 38), 26 percent (29 versus 23), and 9 percent (38 versus 35) for the three cases.

Method 2 is much more efficient than Method 1. The reason for this is that TRANSYT-7F was used only to calculate the PROS for each alternative solution rather than optimizing the offsets. It is expected that if the PROS values were calculated internally without a need to call TRANSYT-7F externally, the time required to run 50 generations (2,500 PROS calculations) would be less than a minute for a 12-intersection artery.

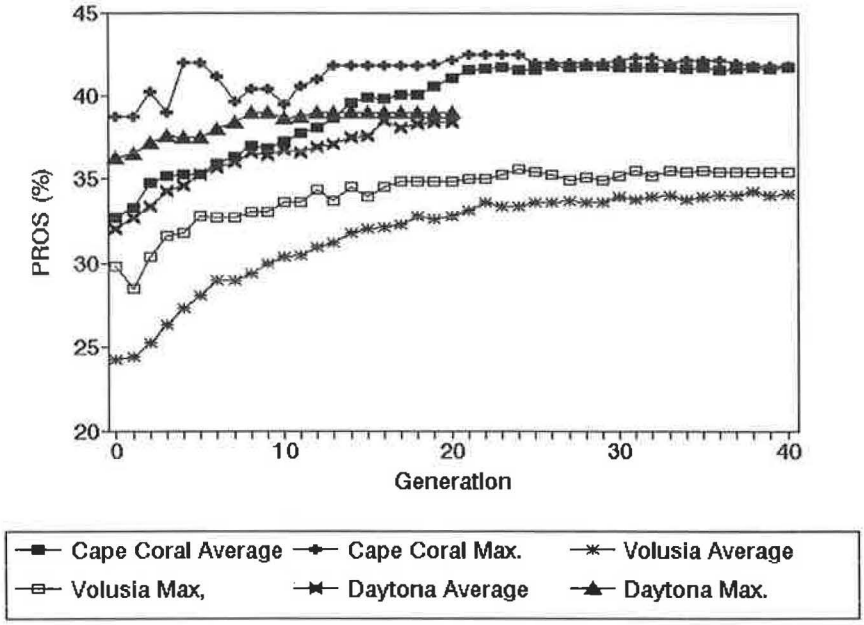


FIGURE 2 Improvement in PROS when optimized using Method 1 with 0.90 crossover probability.

It can be observed from Figure 3, however, that after reaching a certain point in the optimization, the maximum PROS value did not always increase from generation to generation with Method 2. In these cases, the PROS value did not show a clear trend of improvement late in the GA process. This is true for at least two of the three examples and suggests that further improvements might be needed for Method 2.

Table 2 and Figure 4 indicate that using Method 1 or Method 2 to select signal phasing could result in significant improvements in PROS compared with using the existing phase sequences. Method 1 improved PROS by 27 percent (42 versus 33), 44 percent (36 versus 25), and 18 percent (39 versus 33) for the three cases. Method 2 produced 30 percent (43 versus 33), 16 percent (29 versus 25), and 15 percent (38 versus 33) PROS improvements, respectively.

In addition, using Method 1 or Method 2 to optimize phasing produced higher bandwidth efficiency compared with existing phase sequences. For example, Method 1 increased the bandwidth efficiency by 209 percent (34 versus 11) and 138 percent (31 versus 13) for Cape Coral Parkway and Volusia

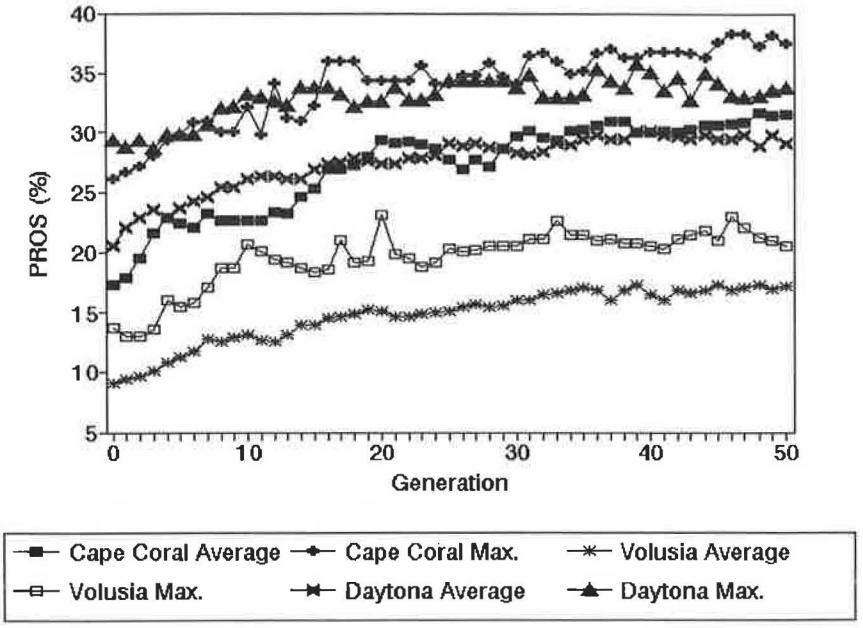


FIGURE 3 Improvement in PROS when optimized using Method 2 with 0.90 crossover probability.

TABLE 2 Comparison of the Results Obtained Using TRANSYT-7F To Optimize PROS with Different Phase Sequences

System	Sequence Source	Artery	Crossover Probability	Effective PROS (%)			Bandwidth Efficiency (%)		
				Right[a]	Left[a]	Average	Right[a]	Left[a]	Average
Cape Coral Parkway	Existing	1	0.90	40	26	33	21	0	11
	PASSER II	1	0.90	31	51	41	22	45	33
	Method 1	1	0.90	38	47	42	27	41	34
	Method 2	1	0.90	35	50	43	27	45	36
	Method 2	1	0.65	42	35	38	38	19	29
Volusia Avenue	Existing	1	0.90	24	25	25	18	7	13
	PASSER II	1	0.90	30	35	32	24	31	28
	Method 1	1	0.90	34	37	36	30	32	31
	Method 2	1	0.90	27	32	29	0	27	14
	Method 2	1	0.65	34	33	34	30	26	28
Daytona Beach Network	Existing	1	0.90	31	37	33	15	28	21
		2		34	28		25	4	15
	PASSER II	1	0.90	45	33	37	45	26	35
		2		31	36		17	33	25
	Method I	1	0.90	42	42	39	33	38	35
		2		32	32		22	18	20
	Method 2	1	0.90	43	42	38	34	38	36
		2		38	22		36	0	18

[a] Right and Left refers to the right-bound and left-bound travel on the artery.

Avenue, respectively. For the two arteries of the Daytona Beach network, Method 1 increased the bandwidth by 66 percent (35 versus 21) and 33 percent (20 versus 15), respectively.

The two GA methods were also compared with using PASSER II for phase sequence optimization on individual arteries to determine whether they could be as effective. The results are shown in Table 2 and Figure 4. Method 1 was able to produce better PROS values in all cases. The improvements achieved were 2 percent (42 versus 41), 12 percent (36 versus 32), and 5 percent (39 versus 37) for the three systems investigated. In addition, Method 1 was able to produce good solutions in terms of bandwidth efficiency relative to PASSER II solutions, as shown in Table 2.

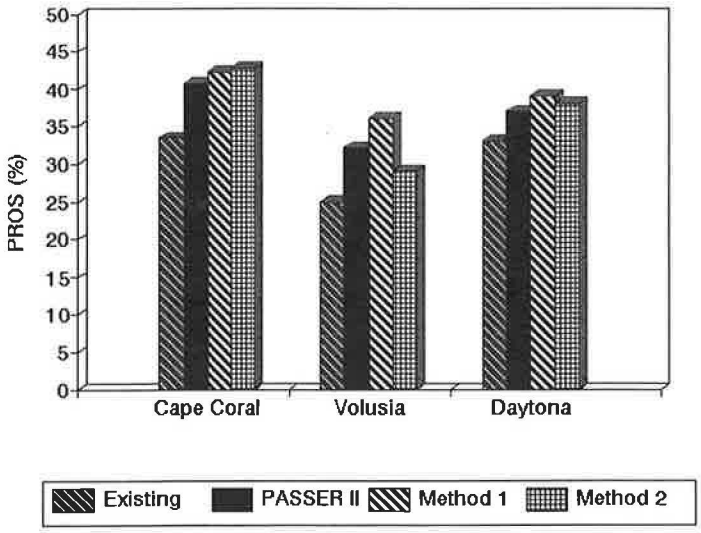


FIGURE 4 Comparison of maximum PROS achieved using different phase sequence sources.

Method 2 was able to select solutions with higher PROS values compared with PASSER II solutions in only two of the three systems investigated (Cape Coral Parkway and Daytona Beach network). In the Volusia Avenue example, Method 2 could not reach as good a solution as that produced by PASSER II, again suggesting that Method 2 needs to be improved.

A sensitivity analysis was performed to examine the effect of varying crossover probability (CP) on Method 2 performance. It was thought that a lower value of this parameter might improve the performance of the model by preventing premature convergence. Table 1 shows that reducing CP from 0.90 to 0.65 increased PROS by 17 percent (34 percent versus 29 percent) for Volusia Avenue; however, it also reduced PROS for Cape Coral Parkway by 10 percent (38 versus 42). This indicated that varying genetic operators and parameters might have important effects on the performance of the model.

CONCLUSIONS AND RECOMMENDATIONS

From the results of this study, it can be concluded that the GA method has potential for use in signal phasing and timing optimization. It appears that the concurrent use of the GA and TRANSYT-7F (that is, Method 1) can optimize the phase sequences for arterial streets as well as multiarterial networks.

Using the GA followed by TRANSYT-7F as in Method 2 can produce as good results as those produced by Method 1 or PASSER II only in some of the cases; however, this method is much more efficient than Method 1 in terms of execution time. Several suggestions for improving the performance of Method 2 might include

1. Varying various GA parameters, including crossover probabilty, mutation probability, population size, string length, and objective function scaling strategy. Lower values of crossover probability might produce better results.
2. Using advanced GA operators.

Implementation of the hybrid GA internally in the TRANSYT-7F program should be investigated further. This will give TRANSYT-7F the ability of simultaneous optimization of all signal timing elements. Further research is needed to reduce the execution time of this strategy, particularly Method 1. Further research is also needed to improve the general optimization process of TRANSYT-7F, because this would also improve its performance with respect to phasing optimization.

REFERENCES

1. Chang, E. C. P., and C. J. Messer. *Arterial Signal Timing Optimization Using Passer II-90—Program User's Manual.* Texas Transportation Institute, Texas A&M University System, College Station, 1991.
2. Wallace, C. E., E. C. P. Chang, C. J. Messer, and K. G. Courage. *PASSER II Users' Guide,* Vol. 3. FHWA, U.S. Department of Transportation, 1991.
3. Wallace, C. E. *MAXBAND Users' Manual.* Transportation Research Center, University of Florida, Gainesville, 1987.
4. Wallace, C. E., K. G. Courage, and M. A. Hadi. *TRANSYT-7F Users' Guide*, Vol. 4. FHWA, U.S. Department of Transportation, 1991.
5. Rogness, R. O., and C. J. Messer. Heuristic Programming Approach to Arterial Signal Timing. In *Transportation Research Record 906,* TRB, National Research Council, Washington, D.C., 1983, pp. 67–75.
6. Cohen, S. L. Concurrent Use of MAXBAND and TRANSYT Signal Timing Programs for Arterial Signal Optimization. In *Transportation Research Record 906,* TRB, National Research Council, Washington, D.C., 1983, pp. 81–84.
7. Cohen, S. L., and C. C. Liu. The Bandwidth-Constrained TRANSYT Signal-Optimization Program. In *Transportation Research Record 1057,* TRB, National Research Council, Washington, D.C., 1986, pp. 1–8.
8. Skabardonis, A., and A. D. May. Comparative Analysis of Computer Models for Arterial Signal Timing. In *Transportation Research Record 1021,* TRB, National Research Council, Washington, D.C., 1985, pp. 45–52.
9. Cohen, S. L., and J. R. Mekemson. Optimization of Left-Turn Phase Sequence on Signalized Arteries. In *Transportation Research Record 1021,* TRB, National Research Council, Washington, D.C., 1985, pp. 53–58.
10. Chang, E. C. P., S. L. Cohen, C. Liu, N. A. Chaudhary, and C. J. Messer. MAXBAND-86: Program for Optimizing Left-Turn Phase Sequence in Multiarterial Closed Networks. In *Transportation Research Record 1181,* TRB, National Research Council, Washington, D.C., 1988, pp. 61–67.
11. Hadi, M. A., and C. E. Wallace. A Progression-Based Optimization Model in TRANSYT-7F. In *Transportation Research Record 1360,* TRB, National Research Council, Washington, D.C., 1992.
12. Hadi, M. A., and C. E. Wallace. Major Enhancements to TRANSYT-7F. Presented at the Fourth International Conference on Microcomputers in Transportation, American Society of Civil Engineers, Baltimore, Md., 1992.
13. Wallace, C. E., and K. G. Courage. *AAP Users' Guide,* Vol. 2. FHWA, U.S. Department of Transportation, 1991.
14. Chaudhary, N. A., A. Pinnoi, and C. J. Messer. Proposed Enhancements to MAXBAND 86 Program. In *Transportation Research Record 1324,* TRB, National Research Council, Washington, D.C., 1991, pp. 98–104.
15. Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, Mass., 1989.
16. Foy, M., R. F. Benekohal, and D. E. Goldberg. Signal Timing Determination Using Genetic Algorithms. In *Transportation Research Record 1365*, TRB, National Research Council, Washington, D.C., 1992.
17. Wallace, C. E., and K. G. Courage. Arterial Progression—New Design Approach. In *Transportation Research Record 881,* TRB, National Research Council, Washington, D.C., 1982, pp. 53–59.
18. Grefenstette, J. J. Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC-16, No. 1, 1986, pp. 122–128.

Publication of this paper sponsored by Committee on Traffic Signal Systems.