Proc. 32nd IEEE Conf. on Decision & Control
San Antonio, Texas - December 1993, pp. 1897-1902

# Self-Organizing Traffic Control via Fuzzy Logic

Stephen Chiu and Sujeet Chand
Rockwell International Science Center
1049 Camino Dos Rios
Thousand Oaks, CA 91360, USA

## Abstract

*We present an approach to self-organizing traffic signal control based on a fully distributed system of cooperative local controllers. The signal timing parameters at each intersection are adjusted by a local controller as functions of the local traffic condition and of the signal timing parameters at adjacent intersections.*

*Each local controller uses a set of fuzzy decision rules to adjust the standard signal timing parameters: cycle time, phase split, and offset. This fully distributed architecture provides for a fault-tolerant, responsive traffic control system, while the underlying fuzzy rule-based algorithm provides for a flexible and easily extensible control law. We show the effectiveness of this method through simulation of the traffic flow in a network of controlled intersections.*

## 1. Introduction

The steady increase in the number of automobiles on the road has amplified the importance of managing traffic flow efficiently to optimize utilization of existing road capacity. High fuel cost and environmental concerns also provide important incentives for minimizing traffic delays. To this end, computer technology has been widely applied to optimize traffic signal timing to facilitate traffic movement.

Traffic signals in use today typically operate based on a preset timing schedule. The most common traffic control system used in the United States is the Urban Traffic Control System (UTCS), developed by the Federal Highway Administration in the 1970's. The UTCS generates timing schedules off-line on a central computer based on average traffic conditions for a specific time of day; the schedules are then downloaded to the local controllers at the corresponding time of day. The timing schedules are typically obtained by either maximizing the bandwidth on arterial streets or minimizing a disutility index that is generally a measure of delay and stops. Computer programs such as MAXBAND [3] and TRANSYT-7F [9] are well established means for performing these optimizations.

The off-line, global optimization approach used by UTCS cannot respond adequately to unpredictable changes in traffic demand. With the availability of inexpensive microprocessors, several real-time adaptive traffic control systems were developed in the late 70's and early 80's to address this problem. These systems can respond to changing traffic demand by performing incremental optimizations at the local level. The most notable of these are SCATS [4,5,8], developed in Australia, and SCOOT [5,7], developed in England.

Both SCATS and SCOOT incrementally optimize the signals' cycle time, phase split, and offset. The cycle time is the duration for completing all phases of a signal; phase split is the division of the cycle time into periods of green signal for competing approaches; offset is the time relationship between the start of each phase among adjacent intersections. SCATS organizes groups of intersections into subsystems. Each subsystem contains only one critical intersection whose timing parameters are adjusted directly by a regional computer based on the average prevailing traffic condition for the area. All other intersections in the subsystem are always coordinated with the critical intersection, sharing a common cycle time and coordinated phase split and offset. At the lower level, each intersection can independently shorten or omit a particular phase based on local traffic demand; however, any time saved by ending a phase early must be added to the subsequent phase to maintain a common cycle time among all intersections in the subsystem. The basic traffic data used by SCATS is the "degree of saturation", defined as the ratio of the effectively used green time to the total available green time. Cycle time for a critical intersection is adjusted to maintain a high degree of saturation for the lane with the greatest degree of saturation. Phase split for a critical intersection is adjusted to maintain equal degrees of saturation on competing approaches. The offsets among the intersections in a subsystem are selected to minimize stops in the direction of dominant traffic flow. Technical details on exactly how the cycle time and phase split of a critical intersection are adjusted are not available from literature. It seems that SCATS does not explicitly optimize any specific performance measure, such as average delay or stops.

SCOOT uses real-time traffic data to obtain traffic flow models, called "cyclic flow profiles", on-line. The cyclic flow profiles are then used to estimate how many vehicles will arrive at a downstream signal when the signal is red. This estimate provides predictions of queue size for different hypothetical changes in the signal timing parameters. SCOOT's objective is to minimize the sum of the average

queues in an area. A few seconds before every phase change, SCOOT uses the flow model to determine whether it is better to delay or advance the time of the phase change by a few seconds, or leave it unaltered. Once a cycle, a similar question is asked to determine whether the offset should be advanced or delayed by a few seconds. Once every few minutes, the cycle time is similarly incremented or decremented by a few seconds. Thus, SCOOT changes its timing parameters in fixed increments to optimize an explicit performance objective.

It is problematic that a specific performance objective will be appropriate for all traffic conditions. For example, maximizing bandwidth on arterial streets may cause extended wait time for vehicles on minor streets. This problem is typically addressed by the use of weighting factors; the TRANSYT optimization program provides user-selectable link-to-link flow weighting, stop weighting factors, and delay weighting factors. A traffic engineer can vary these weighting factors until the program produces a good (by human judgment) compromise solution. In view of the uncertainty in defining a suitable performance measure, the reactive type of control provided by SCATS, where there is no explicit effort to optimize any specific performance measure, appears to have merit. We believe implementing this type of control using fuzzy logic decision rules can further enhance the appropriateness of the control actions, increase control flexibility, and produce performance characteristics that more closely match human sensibility of "good" traffic control.

In work performed by Pappis and Mamdani [6], fuzzy logic was applied to control a lone intersection of two one-way streets. Vehicle detectors were assumed to have been placed sufficiently upstream from the intersection to inform the controller about future arrival of vehicles at the intersection. It is then possible to predict the number of vehicles that will cross the intersection and the size of the queue that will accumulate if no change to the signal state takes place in the next N seconds, for N = 1,2,...10. The predicted outcomes are ranked by fuzzy rules to determine the desirability of extending the current signal state for N more seconds. Each of the possible outcomes is assigned a degree of desirability by the rules, and the extension corresponding to the most desirable outcome is selected for implementation.

Favilla et al. [2] applied fuzzy logic to control a lone intersection of two-way streets. The number of vehicles that have already passed through the green approach and the length of the vehicle queue in the red approach are used as inputs to the fuzzy rules; the output of the rules is the amount of extension to be applied to the current signal state. They also proposed additional strategies for adapting the numerical bounds on the input and output variables.

We have applied fuzzy logic to the general problem of controlling multiple intersections in a network of two-way streets [1]. In our previous work, we proposed a highly distributed architecture in which each intersection is controlled by a local controller that is loosely coordinated with the adjacent intersections. A set of fuzzy rules is used at each intersection to adjust the cycle time, phase split, and offset based on local traffic data collected at the intersection and the timing parameters of adjacent intersections. This architecture provides for a fault-tolerant traffic control system where traffic can be managed by the collective actions of simple microprocessors located at each intersection; hardware failure at a small number of intersections should have minimal effect on overall network performance. In this paper, we extend this approach by presenting improved fuzzy rules that result in faster converging and more effective signal adaptation.

## 2. Fuzzy Control

For completeness, a brief introduction to fuzzy rule-based control is presented in this section. At the basis of fuzzy logic is the representation of linguistic descriptions as *membership functions* [10]. The membership function indicates the degree to which a value belongs to the class labeled by the linguistic description. For example, the linguistic description BIG may be represented by the membership function BIG($x$) shown in Fig. 1, where the abscissa is an input value and the ordinate is the degree to which the input value can be classified as BIG. In this example, the degree to which the number 80 is considered BIG is 0.5, i.e., BIG(80) = 0.5.

Fuzzy control rules are typically expressed in the following form:

$$\text{If } X_1 \text{ is } A_{i,1} \text{ and } X_2 \text{ is } A_{i,2} \text{ then } U \text{ is } B_i \ .$$

where $X_1$ and $X_2$ are the inputs to the controller, $U$ is the output, $A$'s and $B$'s are membership functions, and the subscript $i$ denotes the rule number. For example, a rule for engine control may state "If speed_error is negative_small and speed_error_change is positive_big, then throttle_change is positive_small." Given input values of $x_1$ and $x_2$, the *degree of fulfillment* (DOF) of rule $i$ is given by the minimum of the degrees of satisfaction of the individual antecedent clauses, i.e.,

$$\text{DOF}_i = \text{Min}\ \{A_{i,1}(x_1)\ ,\ A_{i,2}(x_2)\}\ .$$

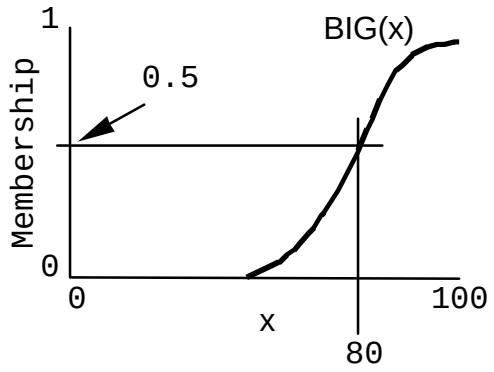


*Fig. 1. Membership function defines a linguistic description.*

We compute the output value by

$$u = \left[ \sum_{i=1}^{n} (\mathrm{DOF}_i) B_i^d \right] \div \left[ \sum_{i=1}^{n} (\mathrm{DOF}_i) \right] .$$

where $B_i^d$ is the *defuzzified* value of the membership function $B_i$, and n is the number of rules. The defuzzified value of a membership function is the single value that best represents the linguistic description; typically, we take the abscissa of a membership function's centroid as its defuzzified value. In essence, each rule contributes a conclusion weighted by the degree to which the antecedent of the rule is fulfilled. The final control decision is obtained as the weighted average of all the contributed conclusions. Although there are several variant methods of fuzzy inference computation, the above method has gained popularity in control applications due to its computational and analytical simplicity.

## 3. Traffic Control Rules

A set of 47 fuzzy control rules was used for adjusting the signal timing parameters. The rules are divided into three decoupled groups: a group of 25 rules for adjusting cycle time and phase split, a group of 18 rules for adjusting offset, and a group of 4 rules for determining appropriate constraints on the cycle time value. The rules for adjusting the cycle time and phase split follow the general principles used by SCATS, except here cycle time and phase split adjustments are coupled to avoid the possibility of one parameter change working against another. The cycle time and phase split are adjusted in concert to maintain good and equal degrees of saturation on competing approaches. The offset at each intersection is adjusted incrementally to coordinate with the adjacent upstream intersection to minimize stops in the direction of dominant traffic flow. We note that if the cycle time of the local intersection is widely different from that of the upstream intersection, then no coordination will be possible; hence a group of rules exists to limit the cycle time difference when coordination is important and allow cycle time to vary freely when coordination is unimportant (i.e., when traffic volume is low). In our previous work [1], we used decoupled rules for each of the cycle time, phase split, and offset adjustments, without any constraint on cycle time differences. The new rules provide more intelligent coordination among parameter adjustments.

### Cycle Time and Phase Split Adjustment

Cycle time and phase split are adjusted in concert to maintain good and equal degrees of saturation on competing approaches. We define the degree of saturation for a given approach as the actual number of vehicles that passed through the intersection during the green period divided by the maximum number of vehicles that can pass through the intersection during that period. Hence, the degree of saturation is a measure of how effectively the green period is being used. The primary reason for maintaining a sufficiently high degree of saturation is not to ensure efficient use of green periods, but to control delay and stops. When traffic volume is low, the green periods must be reduced to maintain a given degree of saturation; this results in short cycle times that reduce the delay in waiting for phase changes. When the traffic volume is high, the green periods must be increased to maintain the same degree of saturation; this results in long cycle times that reduce the number of stops.

The rules for adjusting the cycle time and phase split are shown in the form of a rule matrix in Fig. 2, and the corresponding membership functions are shown in Fig. 3. The inputs to the rules are (1) the highest degree of saturation among the east-west approaches and (2) the highest degree of saturation among the north-south approaches. The outputs of the rules are the amount of adjustment to the current cycle time, expressed as a fraction of the current cycle time, and the amount of adjustment to the current east-west green phase allocation. Because the green phase allocation for each approach is expressed as a percentage of the total cycle time, subtracting from the east-west green allocation will add an equal amount to the north-south green phase.

Each cell in the rule matrix represents a possible combination of input conditions. For example, the upper left cell represents the combination "if east-west saturation is very.low and north-south saturation is very.low". The corresponding output conclusions are contained in each cell, where C denotes the cycle time adjustment, and P denotes the green phase adjustment for the east-west approach. For example, the conclusions in the upper left cell are "cycle time adjustment is negative big and green phase adjustment is zero."

The rules are evaluated at every phase change; the maximum cycle time adjustment allowed in one step is 20% of the

**East-West Saturation**

| North-South Saturation | very.low | low | good | high | very.high |
|---|---|---|---|---|---|
| very.low | C=n.big<br>P=zero | C=n.med<br>P=p.sml | C=n.sml<br>P=p.med | C=zero<br>P=p.big | C=p.sml<br>P=p.big |
| low | C=n.med<br>P=n.sml | C=n.med<br>P=zero | C=n.sml<br>P=p.sml | C=zero<br>P=p.med | C=p.sml<br>P=p.big |
| good | C=n.sml<br>P=n.med | C=n.sml<br>P=n.sml | C=zero<br>P=zero | C=p.sml<br>P=p.sml | C=p.med<br>P=p.med |
| high | C=zero<br>P=n.big | C=zero<br>P=n.med | C=p.sml<br>P=n.sml | C=p.big<br>P=zero | C=p.big<br>P=p.sml |
| very.high | C=p.sml<br>P=n.big | C=p.sml<br>P=n.big | C=p.med<br>P=n.med | C=p.big<br>P=n.sml | C=p.big<br>P=zero |

C = cycle time change
P = east-west green phase change

*Fig. 2. Matrix representing rules for adjusting cycle time and phase split.*

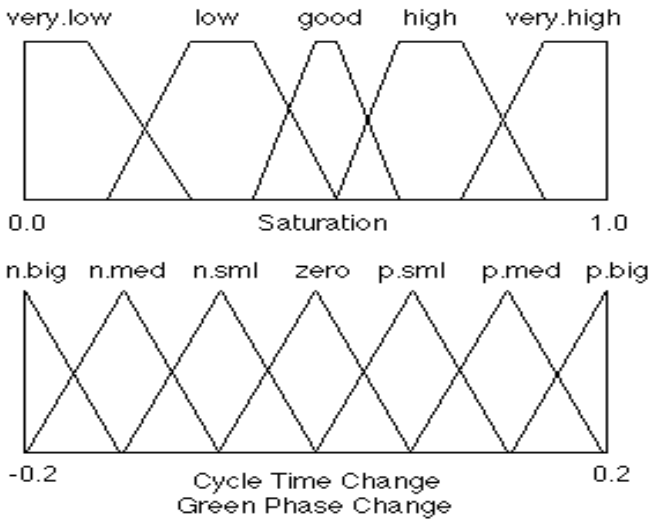


*Fig. 3. Membership functions used in rules.*

current cycle time, and the maximum phase split adjustment allowed in one step is also 20%. The "good" degree of saturation to be maintained by the controller is only 0.55, whereas SCATS typically attempts to maintain a degree of saturation of 0.9. This discrepancy arises from the method of calculating the maximum (saturated) vehicle flow value. We derive the maximum flow value based on a platoon of vehicles with no gaps moving through the intersection at the speed limit, while SCATS uses calibrated, more realistic values.

**Offset Adjustment**

Offset is adjusted to coordinate adjacent signals such that stops in the direction of dominant traffic flow are minimized. The controller first determines the dominant direction from the vehicle count for each approach, and thus determines the upstream intersection with which it wishes to be coordinated. Based on the next green time of the upstream intersection, the arrival time of a vehicle platoon leaving the upstream intersection can be calculated. If the local signal becomes green at that time, then the vehicles will pass through the local intersection unstopped. The required local adjustment to the time of the next phase change is calculated based on this target green time. Fuzzy rules are then applied to determine what fraction of the required adjustment can be reasonably executed in the current cycle. Some of the rules for determining the allowable adjustment are shown in Fig. 4. The inputs to the rules are (1) the difference between the traffic volume in the dominant direction and the average volume in the remaining directions ("vol_diff"); and (2) the required time adjustment relative to the adjustable amount of time ("req_adjust"), e.g., the amount by which the current green phase is to be shortened/extended divided by the current green period. The output of the rules is the allowable adjustment, expressed as a fraction of the required amount of adjustment. These rules will allow a large fraction of the adjustment to be made if a significant advantage is to be gained by coordinating the flow in the dominant direction and if the adjustment can be made without significant disruption to the current schedule.

```
if vol_diff is very.high & req_adjust is low
       then allow_adjust is all;
if vol_diff is very.high & req_adjust is high
       then allow_adjust is medium;
            :
if vol_diff is low & req_adjust is high
       then allow_adjust is low;
if vol_diff is none
       then allow_adjust is none;
```

*Fig. 4. Some rules for adjusting offset.*

**Cycle Time Constraint**

Regardless of how the offset is adjusted, consistent coordination of the local intersection with the upstream intersection is possible only if the two intersections share a similar (not necessarily equal) cycle time. Consider the situation where a local intersection has relatively low traffic volume while the upstream intersection has high traffic volume, then the local intersection will typically have a short cycle time while the upstream intersection will have a long cycle time. Forcing the local intersection to share a similar cycle time with the upstream intersection will usually produce more harm than good; as the local intersection must take on an unnecessarily long cycle time, the extra time spent by vehicles waiting at the intersection will likely outweigh the small benefits gained from the coordination. The benefits gained from enforcing a similar cycle time are realized only when the traffic flowing from the upstream intersection to the local intersection has high volume. Hence, we use a set of simple rules (see Fig. 5) to determine the allowable difference between the local cycle time and that of the upstream intersection: if the volume of traffic from the upstream intersection is high, then the local cycle time can differ from the upstream cycle time by only a small amount (e.g., 10%); otherwise it can adapt freely according to the cycle time adjustment rules. The derived bounds on the cycle time are applied as clipping values after the normal cycle time adjustments are made.

```
if veh_volume is very.high
       then cycle_diff is very.small;
if veh_volume is high
       then cycle_diff is small;
if veh_volume is med
       then cycle_diff is medium;
if veh_volume is low
       then cycle_diff is no.limit;
```

*Fig. 5. Rules for constraining the cycle time difference.*

## 4. Simulation Results

Simulation was performed to verify the effectiveness of the distributed fuzzy control scheme. We considered a small network of intersections formed by six streets, shown in Fig.

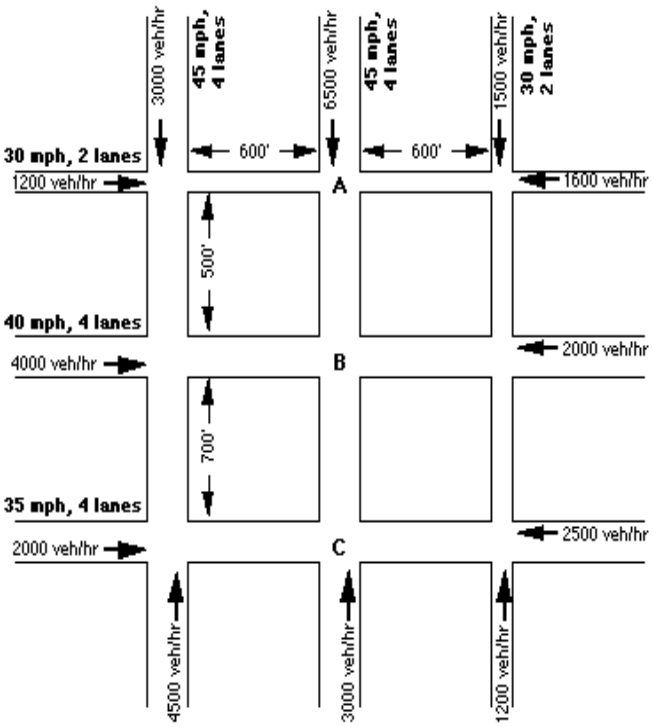


*Fig. 6. Network of streets used in simulation.*

6. A mean vehicle arrival rate is assigned to each end of a street. At every simulation time step, a random number is generated for each lane of a street and is compared with the assigned vehicle arrival rate to determine whether a vehicle should be added to the beginning of the lane. Some simplifying assumptions were used in the simulation model: (1) unless stopped, a vehicle always moves at the speed prescribed by the speed limit of the street, (2) a vehicle cannot change lane, and (3) a vehicle cannot turn. The omission of vehicle dynamics in the simulation tends to result in signal durations that are significantly shorter than those used in the real world; however, the control algorithm does not lose any generality. Vehicle counters are assumed to be installed in all lanes of a street at each intersection. When the green phase begins for a given approach, the number of vehicles passing through the intersection during the green period is counted. The degree of saturation for each approach is then calculated from the vehicle count and the length of the green period. At the start of each phase change, the controller computes the time of the next phase change using its current cycle time and phase split values. The fuzzy decision rules are then applied to adjust the time of the next phase change according to the offset adjustment rules; the adjusted cycle time and phase split values are used only in the subsequent computation of the next phase change time.

Figure 7 shows the average waiting time per vehicle per second spent in the network as a function of time. Figure 8 shows the number of stops per minute encountered by all vehicles. For the first 30 minutes of this simulation, all intersections have a fixed cycle time of 40 seconds, a green duration of 20 seconds, and start their phases at the same time. At the end of 30 minutes, intersections A, B, and C shown in Fig. 6 were allowed to adapt their timing parameters according to the fuzzy decision rules. At the end of 60 minutes, all intersections were allowed to adapt. We see that the improvement in waiting time is minimal when only 3 intersections are adaptive. Furthermore, when only 3 intersections are adaptive, the minor improvement in waiting time was obtained at the expense of greatly increased number of stops. This is because the cycle time chosen by the adaptive intersections (around 15 sec) is widely different from the cycle time for the fixed intersections (40 sec). The mismatch of cycle times resulted in a complete lack of coordination between the adaptive intersections and the fixed intersections, where timing adjustments to facilitate local traffic movement adversely affected the overall traffic movement. (Note that since the dominant traffic flow is through the corridor of A, B, and C intersections, these intersections simply coordinated the local cycle times among themselves, without considering the cycle time of the fixed intersections.) When all intersections were allowed to adapt, all intersections attained similar cycle times (around 15 sec), and significant reductions in both waiting time and number of stops were achieved.

Figures 9 and 10 show the results of a simulation performed using the same sequence of events, but with an initial cycle time of 20 seconds and green duration of 10 seconds for all intersections.

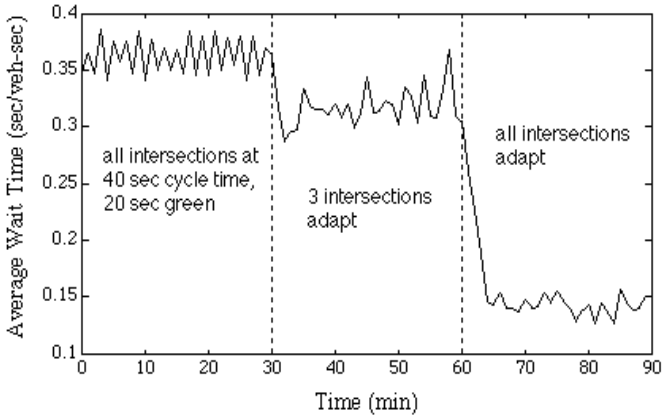


*Fig. 7. Average waiting time for the case in which all intersections have an initial cycle time of 40 seconds.*

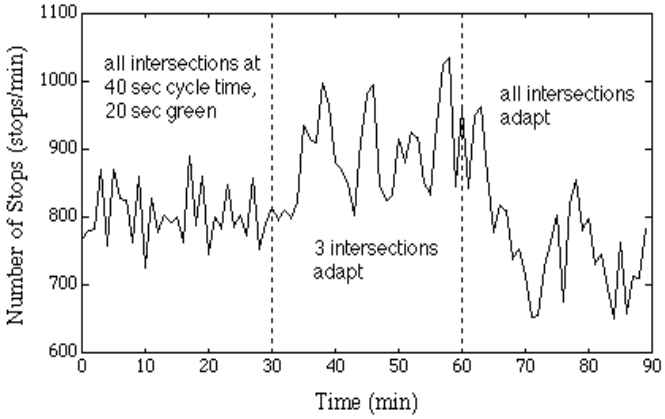


*Fig. 8. Number of stops for the case in which all intersections have an initial cycle time of 40 seconds.*

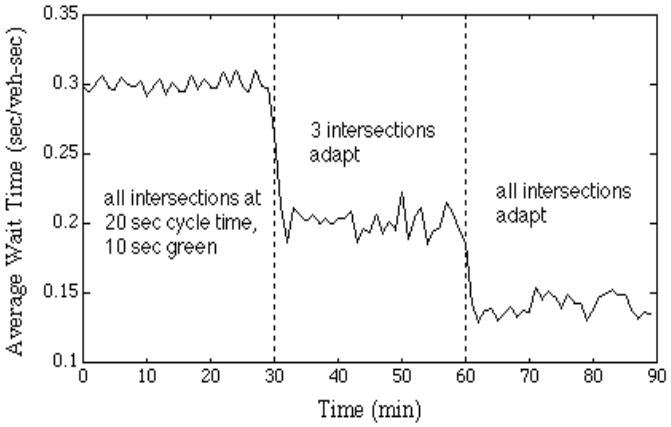


*Fig. 9. Average waiting time for the case in which all intersections have an initial cycle time of 20 seconds.*

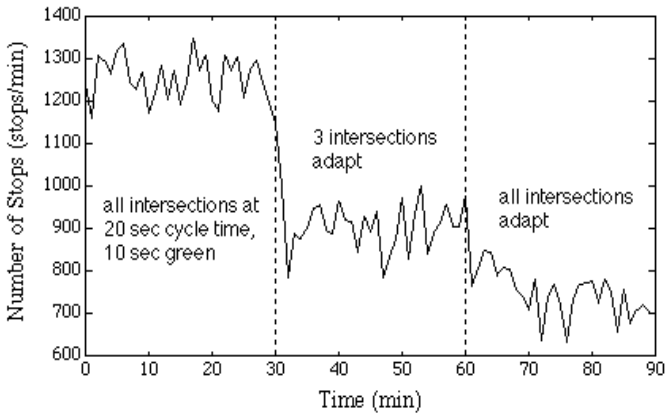


*Fig. 10. Number of stops for the case in which all intersections have an initial cycle time of 20 seconds.*

In this case, significant reductions in both waiting time and number of stops were achieved even when only 3 intersections are adaptive. This is because the cycle time chosen by the adaptive intersections is close to that for the fixed intersections, though not identical. Sharing a similar cycle time has enabled the 3 adaptive intersections to have immediate positive effect on overall system performance.

## 5. Concluding Remarks

We have investigated the use of fuzzy decision rules for adaptive traffic control. A highly distributed architecture was considered, where the timing parameters at each intersection are adjusted using only local information and are coordinated only with adjacent intersections. Although this localized approach simplifies incremental integration of the fuzzy controller into existing systems, simulation results show that the effectiveness of a small number of "smart" intersections is limited if they operate at a cycle time widely different from the rest of the system. In this case, constraining the controller to maintain a cycle time close to the existing system may provide better overall performance.

There is much that can be done to further improve the present fuzzy controller, such as including queue length as an input and using trend data for predictive control. The flexibility of fuzzy decision rules greatly simplifies these extensions.

## References

[1] Chiu, S., and Chand, S., Adaptive traffic signal control using fuzzy logic, *Proc. 2nd IEEE Int. Conf. on Fuzzy Systems*, pp. 1371-1376, San Francisco, CA, March 1993.

[2] Favilla, J., Machion, A. and Gomide, F., Fuzzy traffic control: adaptive strategies, *Proc. 2nd IEEE Int. Conf. on Fuzzy Systems*, pp. 506-511, San Francisco, CA, March 1993.

[3] Little, J., Kelson, M. and Gartner, N., MAXBAND: A Program for Setting Signals on Arteries and Triangular Networks. *Transportation Research Record* 795. National Research Council, Washington, D.C., pp. 40-46, 1981.

[4] Lowrie, P., SCATS - A Traffic Responsive Method of Controlling Urban Traffic. Sales information brochure published by Roads & Traffic Authority, Sydney, Australia, 1990.

[5] Luk, J., Two traffic-responsive area traffic control methods: SCATS and SCOOT. *Traffic Engineering and Control*, pp. 14-20, 1984.

[6] Pappis, C. and Mamdani, E., A fuzzy logic controller for a traffic junction. *IEEE Trans. Syst., Man, Cybern*. Vol. SMC-7, No. 10, 1977.

[7] Robertson, D. and Bretherton, R. D., Optimizing networks of traffic signals in real time - the SCOOT method. *IEEE Trans. on Vehicular Technology*. Vol. 40, No. 1, pp. 11-15, 1991.

[8] Sims, A., The Sydney Coordinated Adaptive Traffic System. *Proc. ASCE Engineering Foundation Conference on Research Priorities in Computer Control of Urban Traffic Systems*, pp. 12-27, 1979.

[9] Wallace, C. et. al., TRANSYT-7F User's Manual (Release 6). Prepared for FHWA by the Transportation Research Center, University of Florida, Gainesville, FL, 1988.

[10] Zadeh, L., Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Trans. Syst., Man, Cybern*. Vol. SMC-3, No. 1, pp. 28-44, 1973.