Proceedings of the 1996 IEEE International
Conference on Control Applications
Dearborn, MI • September 15-18, 1996

WA02 9:30

A Decentralized Scheme for Real-Time Optimization of Traffic Signals†

I. Porche, M. Sampath, R. Sengupta, Y.-L. Chen, and S. Lafortune
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122
porche@umich.edu

Abstract

A real time, highly decentralized, adaptive traffic signal optimization method, ALLONS-D, based on the Rolling Horizon Dynamic Programming technique, is presented in this paper. We describe the basic architecture, the system model, and the optimization scheme of ALLONS-D. We compare ALLONS-D with other approaches to signal control via several sets of simulation results. The margins of improvement are significant when compared with the standard Webster's criteria for signal setting. The feasibility of on-line implementation of ALLONS-D is discussed as well as on-going evaluations and extensions.

1. Introduction

Traffic signal control is a central issue in the design of Advanced Traffic Management Systems (ATMS). Although the first traffic signals were installed in 1914, recent strides in communications and computing power have opened new possibilities for traffic signal control. Traffic control systems seek to minimize the delay experienced by vehicles traveling through a network of intersections. There are various levels of sophistication in traffic control systems. The least complicated is an empirical rule called Webster's Method. It indicates how to divide the amount of green time in a periodic cycle between opposing flows of traffic at an intersection. More advanced traffic control systems anticipate the traffic approaching an intersection. Optimal sequences of green time (plans) are determined using the predicted arrivals of vehicles. Thus, these systems adapt to the traffic demand. Signal plans implemented are selected so that cost (typically, the total delay experienced by the vehicles) is minimized.

†Research supported in part by the University of Michigan Research Center of Excellence in Intelligent Transportation Systems funded by FHWA and industrial affiliates and in part by the National Science Foundation under grant ECS-9057967.

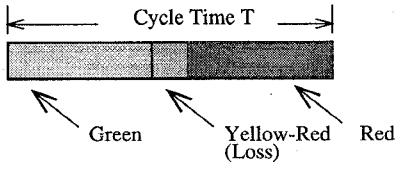

Figure 1: Breakdown of a single fixed cycle

There are two basic types of control for traffic lights at intersections: **fixed-cycle** and **traffic responsive**. Fixed-cycle intersections operate with a constant period of time T. For fixed-cycle schemes, a cycle is divided into a green period, a red period, and a yellow-red time period for each phase. (A phase of traffic is defined as the flows of vehicles that may proceed through an intersection without conflict.) The sequence of green, red and yellow is continuously repeated for each traffic light (see Figure 1). Fixed-cycle intersections can be coordinated by *offsetting* the start of the green time for consecutive intersections along a desired path in order to create what is called a green wave of lights.

An adaptive traffic control system that is decentralized has each intersection determine its signal plan locally. Both decentralized and centralized schemes exist in the field. Most of the early traffic-responsive schemes were based on selecting on-line predefined plans from a set of plans generated off-line [1]. Implementation of these methods involved choosing the most suitable predefined plan and adapting this plan to current traffic conditions. These adaptations involved, for example, extending the green time given to a particular phase or skipping a phase based on current demand. Other schemes simply implemented off-line strategies for signal control at increased frequencies. Overall, these adaptive schemes gave very marginal improvements in performance, if any, and often resulted in poorer performance than fixed-cycle plans [2]. More recent adaptive schemes, such as SCOOT [3] and SCATS [4], however, do not rely on pre-defined plans and are based on on-line optimization schemes. The SCOOT (Split, Cycle and

0-7803-2975-9/96/$5.00 © 1996 IEEE

Offset Optimizer) system attempts an incremental optimization of cycle time, phase split, and offset, and it is based on a platoon dispersion and queue profile traffic model similar to that of the off-line scheme TRANSYT [5]. SCATS (The Sydney Co-ordinated Adaptive Traffic System) optimizes cycle length, phase split, and offset so as to maintain the highest degree of saturation in the traffic network. We refer the reader to [1] for an overview of these schemes.

In this paper we present ALLONS-D, a real-time traffic responsive strategy for decentralized control of traffic signals. ALLONS-D (Adaptive Limited Lookahead Optimization of Network Signals - Decentralized Version) is a *decentralized* method based on *Rolling Horizon Dynamic Programming (RHDP)*. Its objective is to minimize the total delay experienced by all network traffic at each intersection in a given network. The primary feature distinguishing ALLONS-D from schemes such as SCOOT and SCATS is that it is not constrained to generate cyclic plans. Any *arbitrary* phase sequencing and phase splits are permitted subject only to constraints of a minimum green time and a maximum green time for any phase, in the interests of safety and fairness, respectively. ALLONS-D can consider additional priority for vehicles of different types and/or occupancy levels in the traffic stream. *RHDP*, as a strategy for on-line signal control, has been studied by several transportation researchers, and schemes such as OPAC [2, 6], PRODYN [7], and UTOPIA [8] have been developed and tested. ALLONS-D has several features distinguishing it from these methods. We discuss this in more detail later in this paper.

This document is organized as follows. In §2 we describe the basic architecture of ALLONS-D, the system model we utilize, and the decision tree used for optimization. The search algorithm is described in §3. In §4, some preliminary simulation results are presented. Finally, in §5 we comment on the real time implementation of ALLONS-D and we briefly describe on-going extensions of our work.

2. Architecture & Optimization Strategy

The ALLONS-D decentralized architecture consists of one signal controller at each intersection of the network which attempts to minimize the aggregate delay at all approaches to that intersection. In order to generate efficient timing plans, the signal optimizer at each intersection needs information on the current queue lengths and *future arrivals* at its approaches. We assume that whenever the optimizer recalculates the signal plan, which is at a preset time period denoted by Δ, it takes a snapshot of the system and collects arrival information on all of its approaches. Further we assume that the controller looks only as far upstream as the previous intersection. Traffic on links (roads) that are further upstream is ignored. Figure 2 shows a typical snapshot at an intersection.

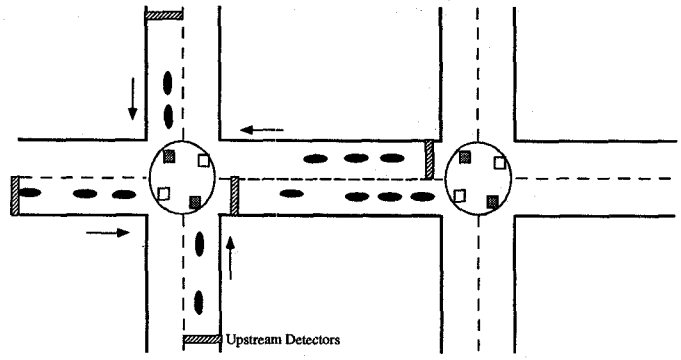

Figure 2: A typical intersection snapshot

The queue lengths and arrival information required by the optimizer could be made available in one of several ways: by upstream detectors located on all approaches to the intersection, by a combination of upstream and downstream detectors, or via surveillance cameras like AUTOSCOPE [9]. The intersection controller requires knowledge of the exact time at which the vehicle arrives at the end of the link and the queue it joins on reaching the intersection. In summary, ALLONS-D has a decentralized, *open* architecture suitable for implementation with existing detector technology, with more sophisticated schemes such as AUTOSCOPE, or, with information available from guided vehicles.

2.1. Decision tree

ALLONS-D computes the optimal signal switching sequence at an intersection using a dynamic programming strategy. The decision that the optimizer needs to take at each point in time is to choose the phase to be green. The number of possible choices at each decision point is equal to the number of phases at the intersection. Furthermore, assume that whenever green time is allocated to a particular phase, this phase remains green until the next time the controller takes a snapshot of the system at which point a new decision is made. Therefore, we see that the decision space has a tree structure and searching the decision space is analogous to building the tree. An exhaustive search of the entire decision space results in a full tree being built. Efficiency in searching the decision space is considered by the degree to which the entire tree will not have to be built to find an optimal path. ALLONS-D is designed to perform a relatively efficient search for an optimal path. This will be explained later in the paper. For a two-phase intersection, the decision space is a binary tree as shown in

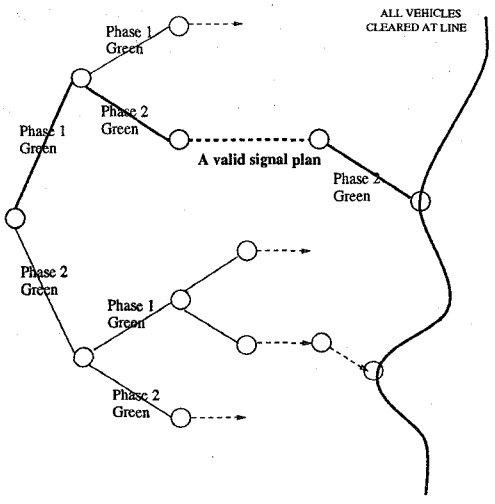

Figure 3: Example decision tree

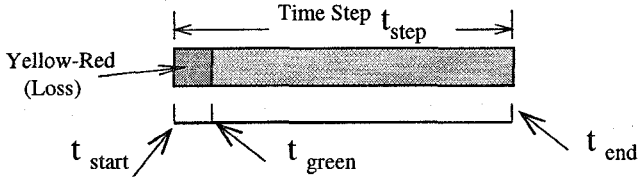

Figure 4: Epochs in a time step

Figure 3.

A valid signal plan is traced from the initial starting node at the root of the decision tree to a node where the intersection is cleared, which we refer to as an end node. Multiple valid signal plans are compared to find one with minimal delay. Delay calculations are described in §2.3 . In general, each branch of the tree represents an interval of time that a particular phase will have a green light. This interval is known as a time step (t_{step}). If a light is continuing green then the time step is the delta time ($t_{step} = \Delta$). If the light has just changed to green then it is the minimum green time plus the yellow-red time ($t_{step} = t_{MG} + t_{YR}$). During a yellow-red (loss) period, no vehicles can leave the intersection, thus a portion of the time step does not allow departures. A time step t_{step} can be further subdivided between three boundaries of time as shown in Figure 4.

2.2. The System Model

The beginning and end of each time interval (or branch) in the decision tree defines a node. These nodes, shown in Figure 3, represent the state at an intersection. A state at an intersection, denoted by the vector $\bar{X}$, is defined by the following information: (1) the absolute time at that node in the decision tree (t), (2) the queue sizes on all approaches ($\bar{q}(t) = [q^1 \cdots q^\ell]'$), (3) the current green phase ($p(t)$), and (4) the time the current green phase has been active ($g(t)$). Hence, $\bar{X}$ has four state components x_1, a vector $\bar{x}_2$, x_3, and x_4. The vector $\bar{x}_2$, the queue sizes, has a row size equal to the number of lanes approaching the intersection. The state is again summarized in Equation 1 below.

$$\bar{X}[d] = \begin{bmatrix} x_1[d] \\ \bar{x}_2[d] \\ x_3[d] \\ x_4[d] \end{bmatrix} = \begin{bmatrix} t \\ \bar{q}(t) \\ p(t) \\ g(t) \end{bmatrix}. \quad (1)$$

Starting from the root, each node expansion implies a search at an increased level of depth. Using d to denote the search depth, the root is set at search depth zero ($d = 0$). Any two nodes described by the same number of queued vehicles, the same green phase, and the same duration of green for that phase at the same time t are the same node and thus the same state. The state of an intersection is updated by extending the current green phase or changing to another green phase. This is equivalent to increasing the depth of the search ($d \leftarrow d + 1$) and continuing along a path to another node. This is characterized by the following non-linear state update equation:

$$\bar{X}[d+1] = f\left(\bar{X}[d],\ u[d],\ A(\cdot)\right). \quad (2)$$

The control variable is $u(\cdot)$. It can take on certain (admissible) integer values including zero. This variable reflects a decision to either change to make another phase p green ($u(\cdot) \neq 0$) or to maintain the current phase as green ($u(\cdot) = 0$) for a duration of time. The vector variable $A(\cdot)$ represents vehicle arrivals within a certain interval of time. This vector consists of arrivals on each approach of the intersection and is an external input to the system. Only arrivals within a specified time interval are considered; this time interval is dictated by the control $u(\cdot)$, the existing state $\bar{X}(\cdot)$, and the time t. Note that the total number of arrivals considered is that obtained when the "snapshot" of the system is taken and it is geographically bound by the neighboring intersection (see Fig. 2). The update equations for each state component are as follows:

$$x_1[d+1] = x_1[d] + t_{step} \quad (3)$$

$$\bar{x}_2[d+1] = \bar{x}_2[d] - D(\bar{x}_2[d], u[d], A([t, t_{step}])) \quad (4)$$

$$x_3[d+1] = x_3[d] + u[d],\ u[d] \text{ is admissible}^1 \quad (5)$$

$$x_4[d+1] = \begin{cases} t_{MG} & \text{if } u(\cdot) \neq 0 \\ x_4[d] + t_{step} & \text{else.} \end{cases} \quad (6)$$

The number of departures $D(\cdot)$ (which could be negative) is a function of the control variable, the prior

[1]For example, at an intersection with two phases 1 & 2, if the current phase is 1, then the admissible control is either 0 of 1. If $u(\cdot) = 1$ is applied when $x_3[d] = 1$, then $x_3[d+1] = 2$.

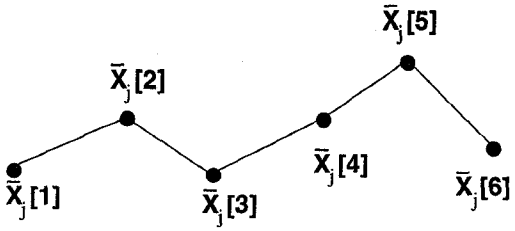

Figure 5: A single path

queue sizes ($\bar{x}_2$), and the arrivals during the time interval. Note how the parameter t_{step} is updated by the control:

$$t_{step} = \begin{cases} \Delta & \text{if } u(\cdot) = 0 \\ t_{MG} + t_{YR} & \text{otherwise.} \end{cases} \quad (7)$$

Consider the depth-first expansion of nodes along the j^{th} path as shown below in Figure 5. A path is formed by any sequence of possible decisions. Consider the *trace* of a path as a sequential list of the phases (i.e., 1 2 1 1 2 ...). For two paths i and j and two possibly different depths d^1 and d^2, if $\bar{X}_i[d^1] = \bar{X}_j[d^2]$ then these two states are equivalent and the graph would no longer be a tree but a decision network with more than one path to at least one node. In ALLONS-D, all decisions are built in a tree like fashion. Even though some nodes may be identical (thus resulting in a network), the computational requirements to compare each newly-generated node to all existing nodes could be burdensome. We chose not to do such comparisons and thus treat the decision network as a tree. The search algorithm used for this tree is described in §3.

A cost can be associated with each node along a path. The cost at a node is the cost to reach the node from the root. Subsequent decisions from this node result in an additional *arc cost* that is incurred based on the decision $u(\cdot)$ corresponding to that arc. Clearly, the *arc cost* is the cost difference between a parent and a child node. The cost of a path is equivalent to the cost at the node at which the path terminates. We now elaborate on the cost function employed in ALLONS-D.

2.3. Delay Calculations

The cumulative delay of all vehicles passing through an intersection is the objective function of the dynamic program. Calculating delay is complex. In OPAC, the number of cars in a queue multiplied by the interval of time in which the cars were queued is considered the cost to be optimized. In ALLONS-D, the sum of the weighted time delay for each vehicle is considered. This allows increased priority to vehicles of a certain type or occupancy by increasing the potential delay that these vehicles may contribute if slowed or stopped.

Each vehicle at an intersection can contribute to delay when the light is red, green, or yellow. Any vehicle stopped at a red light contributes an amount of delay equal to the duration of the red light (multiplied, for this case and the ones below, by the weight associated with the vehicle). In addition, delay can be accumulated by vehicles at a green light. For instance, for long (standing) queues, only a small portion in the front of the queue will clear the intersection when the light turns green. Even for the cars that are close enough to the intersection to pass through, there will still be some delay incurred. The latter type of delay accounts for the minimum amount of time required for the vehicle immediately in front to respond and move through the intersection. This minimum time required for each car will be referred to as the *headway constraint.* It is calculated using the saturation flow (f^{ℓ}_{sat}) for a link ℓ. Hence, delay incurred by vehicles on a link with a green light is dependent on the maximum number of vehicles that could leave during a given time interval. This is dependent on the saturation flow and the length of the interval of green light ($t_{end} - t_{start}$). The saturation flow and the maximum number of vehicles is link specific.

In summary, there are five different contributions to delay: (1) A vehicle waits at a red light, (2) A vehicle is at a light that turns green but cannot leave during the green period due to vehicles ahead of it, (3) A vehicle that is already at a light that turns green leaves after waiting for vehicles ahead of it to depart, (4) A vehicle *arrives during* a green period and will not leave due to vehicles ahead of it, and (5) A vehicle *arrives during* a green period and will leave but is delayed by other vehicles ahead of it. For each case, there is a corresponding delay equation; these equations are not presented here due to space limitations; they can be found in [PL95].

3. ALLONS-D: The Optimization Algorithm

3.1. The STLC Backtracking Procedure

In this section we discuss the search strategy adopted in ALLONS-D to compare different policies and obtain an optimal path in the decision tree. The search is a variation of backtracking [10] which is a depth-first search procedure. A tree is searched by continuously exploring nodes at increasing depths until an end node is reached. An end node is a state where the stopping criterion is met (i.e. $q^{\ell}(t) = 0$) for all links ℓ and there are no remaining arrivals; i.e., the intersection is cleared.

An efficient means to search the tree based on an intuitive knowledge of traffic operations has been devised. This relies on the notion that the policy "serve the

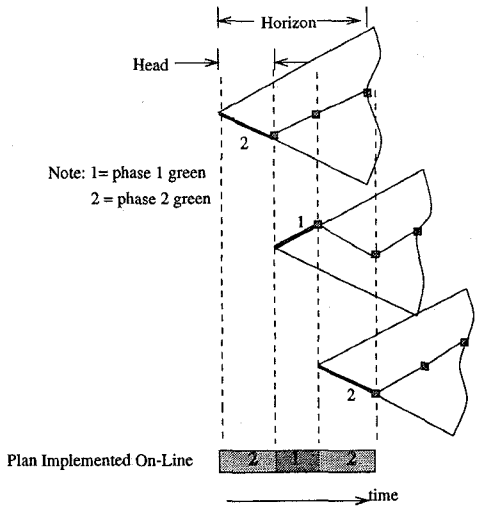

Figure 6: Rolling Horizon: Plan results from several rolls

largest cost" (denoted STLC hereafter) is a "good" policy. STLC is a method of choosing between two or more phases based on the existing cost of the most recent control decision. In other words, the phase that most recently incurred the highest cost or most delay should get the green light. Thus, an initial policy is developed by switching whenever another phase has incurred a higher cost. This continues until the intersection is cleared of all vehicles. This first policy is the first path in the decision tree. The cost of this STLC path and any subsequently found complete paths is compared to maintain a minimum path cost value. From this point in the tree, backtracking is employed to search branches emanating from the STLC path. The motivation behind the STLC heuristic is that an optimum policy may lie near or on this path. If while expanding a node along a path, the path cost exceeds the minimum path cost value, further exploration of the tree from this node will cease. Thus the tree is effectively *pruned*. The process continues until no unexplored nodes are encountered as the algorithm backs into the starting node.

As in OPAC, SPPORT [11], and other strategies, signal plans found by ALLONS-D are implemented using the rolling horizon concept. Once an optimal policy is computed, ALLONS-D implements only the first decision of that policy; this decision is valid for the corresponding t_{step} value of the arc out of the root node of the decision tree. At the end of this time period ALLONS-D updates its arrival information using the most recent detector data, repeats the optimization process over the new decision tree, and computes an optimal policy for the resulting decision tree, and again implements only the first decision. Thus, the horizon is rolled forward one step at a time. This is demonstrated in Figure 6.

3.2. Comparison with Other RHDP Approaches

The enumeration and search of the decision space, as described in §2.1 and §3.1, is more comprehensive than in OPAC since every possible policy is considered. Yet, it is computationally tractable. There lies the potential advantage of ALLONS-D over a scheme like OPAC. OPAC-3, the *RHDP* version of OPAC, uses the notion of sequential constrained search to pick the optimal policy. Namely, the number of switchings during one horizon is constrained to be at least one and at most three. No such constraints are imposed in ALLONS-D. More precisely, of course, there are conditions under which the entire tree would be built but this is almost never the case. The depth-first search in ALLONS-D avoids, in most cases, exhaustive evaluation of all possible decisions. The detailed experimental results presented in §4 suggest that this search technique is computationally implementable in real-time.

In further comparing ALLONS-D to OPAC, it should be noted that ALLONS-D has no predetermined temporal horizon for its lookahead. The horizon is taken to be the time to clear all vehicles currently present on all approaches to the intersection and hence is determined by the current traffic conditions as reported by the upstream detectors. In UTOPIA and PRODYN, two other *RHDP* approaches, each intersection controller takes into account the cost incurred at neighboring intersections calculated on the basis on their last optimized plans. This way coordination may be achieved between adjacent intersections. ALLONS-D, as described in this paper, does not explicitly attempt such coordination due to its highly decentralized nature but rather has a more implicit form of coordination.

4. Simulation Results

An extensive set of simulations was performed to evaluate the performance of ALLONS-D. These simulations involved networks of different topologies and sizes and different sets of travel demands (both constant and time-varying). Due to space limitations, only a subset of the simulation results that have been obtained are presented here; readers interested in further simulation results should contact the authors. Simulations were performed using the commercially available simulation program *INTEGRATION* © Ver. 1.5e [12], as well as a special simulation program called *STNS* [13]. *STNS* was developed at the University of Michigan to function similarly to *INTEGRATION*. Unlike *INTEGRATION* and another well know simulation program, *TRAF-NETSIM*, *STNS* is capable of implementing *RHDP* traffic control algorithms. During a simulation run of *STNS*, the ALLONS-D module freezes the simulator at intervals of Δ seconds and computes the

signal plan as explained in §3. For each intersection in the network, a phase is selected to turn green and the simulation is allowed to run for the next Δ seconds. The whole process then repeats again.

Since our objective is to produce traffic signals that are very responsive to (time-varying) demands, the value of Δ should be chosen small enough to ensure that goal. We have used values of Δ in the range from 5 seconds to 15 seconds in our experiments. For the experiments reported in this section, $\Delta = 5$ seconds. The vehicle arrival information necessary for the ALLONS-D module is directly obtained from the *STNS* simulator which assigns a scheduled end-of-link arrival time to each vehicle entering a link based on a travel time function also used by *INTEGRATION*. The delay experienced by a vehicle at an intersection is the difference between its end of link arrival time and its actual departure time. Delay is a component of cumulative or total travel time of all vehicles traversing links in the network. Hence total travel time will be the primary measure of effectiveness (*MOE*) in these tests. The output produced by *STNS*, in particular "travel time", was compared with that produced by *INTEGRATION* for identical networks, demands, and signal plans. These comparisons are shown in columns 2 and 3 of the data tables for tests of a single intersection, arterial, and triangle network. The data for column 2 is the travel time in *INTEGRATION* resulting from the Webster's optimization performed for the given demands. Column 3 is the travel time reported by *STNS* for the same signal policy and demand. By comparing these two columns, it is evident that both simulators compute this *MOE* in a consistent fashion within a small error. The signal plans in the *INTEGRATION* program are optimized based on the method of Webster and Cobbe [14]. Column 4 represents the travel time in *STNS* for a different signal plan computed using the ALLONS-D algorithm.

At the beginning of each simulation, we assume that there are no vehicles in the simulated network. Further, we do not make any assumptions on the arrival process. Some of the simulations performed assume time varying arrival rates. The simulation time is always set to a value sufficient to allow the network to be cleared. Overall, the results illustrate that the ALLONS-D algorithm develops, in a computationally feasible manner, signal plans that outperform any fixed-cycle plan or adaptive implementation of cyclic plans (*i.e.*, Webster's Method). This is true over a range of possible traffic conditions, from lightly loaded to over-saturated conditions.

4.1. Single Intersection

This subsection describes simulations of a single intersection that is approached by two one-way, one mile long roads. There are two Origin-Destination (O-D)

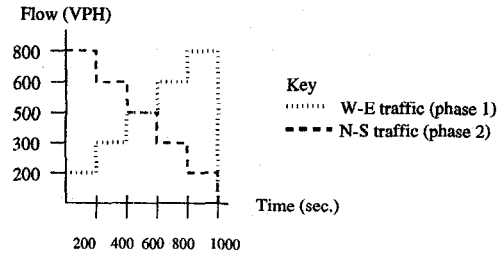

Figure 7: Time varying load for single intersection

pairs in this simple network: W $\leftrightarrow$ E and N $\leftrightarrow$ S. The saturation flow rates on all roads are 1200 vehicles per hour (vph). Seven of the tests consisted of loading each of the two approaches with the same uniform demand for 10 minutes. Starting with 200 vehicles per hour (vph) on each of the two approaches to the intersection for the first test, the demand was incrementally increased for each of six subsequent tests up to 750 vph. The time varying load depicted in Figure 7 is included as the eighth test.

	Col.2 *INTEGRATION* (FC) travel time	Col.3 *STNS* (FC) travel time	Col.4 *STNS* ($\mathcal{A}$) travel time	Col.5 *STNS* ($\mathcal{A}$) delay	Col.6 *STNS* (FC) delay
test	(min.)	(min.)	(min.)	(min.)	(min.)
200	342	336	333	14.9	18.5
300	529	513	506	22.6	32.1
400	735	716	694	32.1	52.8
500	963	933	900	47.8	78.9
600	1495	1554	1110	52.7	388
700	2536	2763	1375	63.6	1255
750	3135	3279	1500	72.5	1658
t.v.	800	795	702	28.6	70

For the tests with uniform demands, the Webster method dictates a fixed, cyclic plan (FC). This was confirmed by the *INTEGRATION* software. The results in the table indicate that ALLONS-D ($\mathcal{A}$) achieves total travel time at least as good as the best fixed plan found using the *INTEGRATION* software. A redundant comparison strictly within the domain of *STNS* indicates the same and illustrates the compatibility between *INTEGRATION* and *STNS* for travel time calculation. The two plans are compared with respect to delay in columns 5 & 6. Note that demands of 600 vph and above for both approaches to the intersection create saturated and over-saturated conditions for which the best cyclic plan is easily outperformed. For a time varying demand, ALLONS-D outperforms any plan computed by *INTEGRATION* which is using Webster and adaptively re-optimizing every 60 seconds. The internal constraints for minimum green time and maximum green

time for the *ALLONS-D* algorithm were made identical to these constraints in the *INTEGRATION* software.

4.2. Arterial

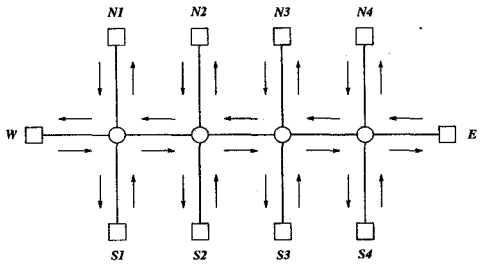

Figure 8: The arterial network.

Three different demands were applied to the arterial network in Figure 8. Each link in the arterial is 1500 feet long. Vehicles are routed without turns. The first test has a constant demand for the W $\leftrightarrow$ E and N $\leftrightarrow$ S routes. Two additional follow-up tests consider time varying demands. These demands are shown in Figure 9. For example, the demand in the second test alternated between 500 and 250 vph for twenty minutes for W $\rightarrow$ E traffic and remained constant on the side streets (N $\leftrightarrow$ S) for an equal amount of time. The best plan found using the *INTEGRATION* software is compared to the best plan using the ALLONS-D algorithm.

By comparing columns 2 & 4, the ALLONS-D algorithm outperforms the best plan found by an adaptive implementation of Webster's method, where a new phase split determined every 60 seconds. Comparing columns 5 & 6 indicates the decreased delay (within the STNS domain) that signal policies determined by ALLONS-D experience relative to signal policies based on Webster's method. Tests 2 & 3 have time varying demands and the degree of variance in the demand is greatest in test 3 where the increase in performance is most profound. Note that the column 2 data was obtained from *INTEGRATION* and the column 3 data from *STNS*.

	Col.2 *INTEGRATION*	Col.3 *STNS*	Col.4 *STNS*	Col.5 *STNS*	Col.6 *STNS*
	(Webster) travel time	(Webster) travel time	($\mathcal{A}$) travel time	($\mathcal{A}$) delay	(Webster) delay
Test	(min.)	(min.)	(min.)	(min.)	(min.)
1	729	717	620	83	190
2	1271	1245	1073	149	353
3	2002	2028	1615	328	746

4.3. Triangle

Three simulations with a triangle network (Fig. 10) are described in this section. The first test had a constant load of 600 vph on three routes (1,4 & 5) and 300 vph on the other three (2,3, & 6). Two other tests

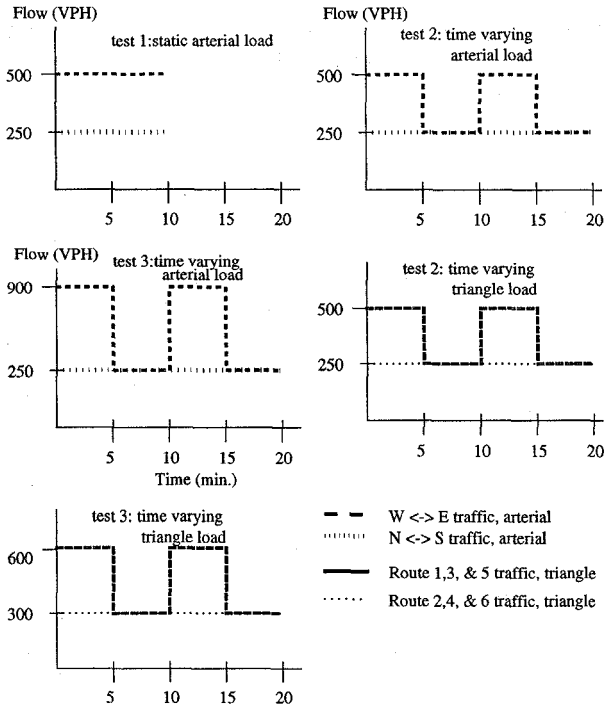

Figure 9: Loads for tests 1-3 on arterial & triangle network

with time varying loads (Fig. 9) substantiate the single intersection and arterial network simulation results. For both the arterial and triangle networks, the Webster optimization module of *INTEGRATION* does not attempt any coordination between intersections via offsets. This would improve performance but probably not to the extent that a relative improvement compared to ALLONS-D results would change our conclusions.

	Col.2 INTEGRATION	Col.3 *STNS*	Col.4 *STNS*	Col.5 *STNS*	Col.6 *STNS*
	Webster travel time	Webster travel time	($\mathcal{A}$) travel time	($\mathcal{A}$) delay	Webster delay
test	(min.)	(min.)	(min.)	(min.)	(min.)
1	2013	2031	1220	160	1000
2	844	931	756	95	289
3	1323	1412	944	114	622

5. Conclusions and Future Work

ALLONS-D, as an *RHDP* approach for demand-responsive signal control, is eminently suited for real-time implementation. We claim this because of the computational speed, decentralization, and simplicity of instrumentation. In practice, we envisage one signal controller at each signalized intersection. The communication infrastructure requirements are minimal. The upstream and downstream detectors, or the

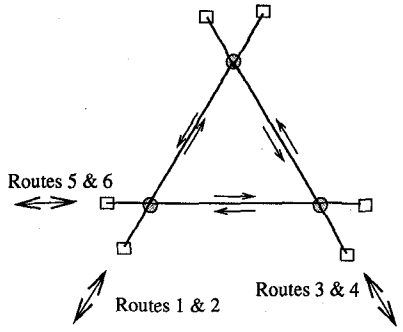

Figure 10: The triangle network.

surveillance cameras, have to be connected to the intersection controller. Other than this the controllers operate autonomously. Simulation results for under-saturated and over-saturated intersections and networks have been presented showing that ALLONS-D compared well with the traditional Webster's approach. In these simulations, ALLONS-D outperforms Webster's method by a large margin in most cases, especially in the cases of heavy traffic loads and time-varying traffic. Comparisons to other systems like SCATS or SCOOT are not possible as they are proprietary, commercial products. (However, SCATS relies heavily on the adaptive Webster implementation enabled in the *INTEGRATION* software in addition to heuristics for coordination.) The incorporation of bus priority is allowed by ALLONS-D and simulation tests have been reported for these cases [15]. We are currently working on comparing ALLONS-D with the traffic actuated policy incorporated in the TRAF-NETSIM software.

The backtracking strategy for optimization for ALLONS-D works well but we are exploring other techniques such as the A* algorithm to improve the efficiency of exploring the decision tree while still preserving optimality. We are also investigating the application of the ALLONS paradigm to coordinated intersection signal control. There is an implicit coordination among intersections in the ALLONS-D architecture due to the incorporation of data from upstream, arrival predicting detectors and the *RHDP* implementation with a small value for the parameter Δ; such as the value $\Delta = 5$ *sec.* used in the experiments presented in §4. The explicit coordination of intersections we seek is via an off-line iterative technique. Finally, a coupling of this traffic control procedure with a dynamic route guidance system is of great interest.

Acknowledgments

The authors acknowledge Yawar Murad for the development of *STNS* used for simulation experiments described in this paper, and Adrian Vlcko for helping to perform these simulations.

References

[1] M. G. Bell, "Future directions in traffic signal control," *Transport Research:A*, vol. 26A, no. 4, pp. 303–313, 1992.

[2] N. H. Gartner, "Demand responsive traffic signal control research," *Transport Research:A*, vol. 19A, pp. 369–373, Feb. 1985.

[3] D. I. Robertson and R. D. Bretherton, "Optimizing networks of traffic signals in real-time – the SCOOT method," *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 11–15, Feb. 1991.

[4] P. Lowrie, "SCATS: A traffic responsive method for controlling urban traffic," tech. rep., Roads and Traffic Authority, NSW, Australia, Traffic Control Section, P.O.Box 693, Darlinghurst, NSW 2010, Australia, 1990.

[5] D. I. Robertson and P. Hunt, "A method of estimating the benefits of coordinating signals by TRANSYT and SCOOT," *Traffic Eng. Control*, 1982.

[6] "Evaluation of the optimized policies for adaptive control strategy," Tech. Rep. FHWA-RD-89-135, Federal Highway Administration, 6300, Georgetown Pike, McLean, Virginia 22101-2296, 1989.

[7] J. J. Henry, J. L. Farges, and J. Tuffal, "The PRODYN real time traffic algorithm," in *4th IFAC-IFIP-IFORS Conference on Control in Transportation System*, (Baden-Baden, Germany), Sept. 1983.

[8] V. Mauro and C. DiTaranto, "UTOPIA," in *6th IFAC-IFIP-IFORS Conference on Control, Computers, Communication in Transport*, (Paris), Sept. 1989.

[9] P. G. Michalopoulos, "Vehicle detection video through image processing: The autoscope system," *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 21–29, Feb. 1991.

[10] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

[11] S. Yagar and B. Han, "A procedure for real-time signal control that considers transit interference and priority," *Transportation Research-B*, vol. 28B, pp. 315–331, Aug. 1994.

[12] M. VanAerde, J. Voss, A. Ugge, and E. R. Case, "Managing traffic congestion in combined freeway and traffic signal networks," *ITE Journal*, pp. 36–42, Feb. 1989.

[13] Y. Murad and S. Lafortune, "STNS: A Simple Traffic Network Simulator," Tech. Rep. CGR-95-13, College of Engineering Control Group Report Series, The University of Michigan.

[14] F. Webster and B. Cobb, *Traffic Signal Settings*. London: H.M.S. Office, 1966. Road Research Technical Paper No. 56.

[15] I. Porche and S. Lafortune, "Dynamic traffic control: Decentralized and coordinated methods using a limited lookahead strategy," Tech. Rep. CGR-95-12, College of Engineering Control Group Report Series, The University of Michigan, Dec. 1995.