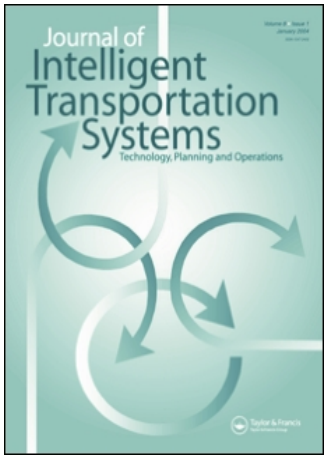

Journal of Intelligent Transportation Systems
Technology, Planning, and Operations
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713398522

Adaptive Look-ahead Optimization of Traffic Signals

Isaac Porche [a]; Stéphane Lafortune [a]
[a] Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI, USA

Online Publication Date: 01 January 1999
To cite this Article: Porche, Isaac and Lafortune, Stéphane (1999) 'Adaptive Look-ahead Optimization of Traffic Signals ', Journal of Intelligent Transportation Systems, 4:3, 209 - 254
To link to this article: DOI: 10.1080/10248079908903749
URL: http://dx.doi.org/10.1080/10248079908903749

ITS Journal, 1999, Vol. 4, pp. 209–254
Reprints available directly from the publisher
Photocopying permitted by license only

Adaptive Look-ahead Optimization of Traffic Signals*

ISAAC PORCHE and STÉPHANE LAFORTUNE†

Department of Electrical Engineering and Computer Science, The University of Michigan, 3300 EECS Building, Ann Arbor, MI 48109-2122, USA

(Received 11 August 1998)

ALLONS-D is a decentralized real-time traffic control scheme. A description of its basic architecture, system model, and optimization algorithm is given. Through simulations, the scheme is shown to be suitable for adaptive control of an isolated intersection. An efficient tree searching algorithm is used to find a traffic signal plan that minimizes delay. The search is constrained only by maximum and minimum green times per phase. This algorithm is free to consider cyclic and non-cyclic signal plans alike. The rolling horizon concept is used for the calculation and implementation of optimal signal plans. The ALLONS-D scheme has several key differences with other real-time traffic control schemes, such as OPAC. Extensive simulation experiments on a single intersection indicate the success of the ALLONS-D scheme versus any fixed-cycle plan. The tests included a full range of uniform and non-uniform demands from low levels of saturation to the over-saturated case. For arterial networks of intersections, the scheme is shown to be capable of inducing a level of coordination that results in progressive signal plans. In particular, the scheme can be used to coordinate travel times on transit routes. An extension of the scheme to provide network-wide optimization of traffic signals is explained in the form of a two-layer scheme. Some preliminary results using this approach are included. Simulation experiments are performed with a newly developed-traffic simulator called STNS.

Keywords: Intelligent Transportation Systems; Real-time adaptive traffic signal control; Coordinated traffic signals; Transit priority systems

* Research supported in part by the Rackham Merit Fellowship Program, and by the University of Michigan Research Center of Excellence in Intelligent Transportation Systems funded by FHWA and industrial affiliates.

† Corresponding author. E-mail: porche@umich.edu, stephane@umich.edu.

INTRODUCTION

There are various levels of sophistication in traffic signal control systems; quoting from (DOT, 1996): "Current state-of-the-art traffic signal control systems have the capability to dynamically modify the signal timings in response to changing traffic demand and to coordinate operation between adjacent signals to maximize the roadway (network) throughput... At a minimum, these coordinated signal control systems provide for (...) special signal timing patterns that can optimize operations along major arterial routes and over traffic networks."

This paper is concerned with these types of adaptive traffic control systems that anticipate the arrivals of vehicles approaching an intersection. Such systems calculate optimal sequences of green time to direct the competing traffic flows at the intersection. These sequences are referred to as signal plans. We presume that these arrivals of vehicles are determined using detection devices and traffic models sufficient to allow the signal control to adapt to the traffic demand. Furthermore, signal plans implemented by such systems are selected so that cost (typically, the total delay experienced by the vehicles) is minimized.

The main effort of this paper is to describe ALLONS-D (Adaptive Limited Lookahead Optimization of Network Signals – Decentralized), a real-time traffic responsive strategy for decentralized control of traffic signals. ALLONS-D is a *decentralized* method based on the *Rolling Horizon* (*RH*) concept in dynamic optimization. Its objective is to minimize the total delay experienced by traffic at each intersection in a given network. The primary feature distinguishing ALLONS-D from schemes such as SCOOT (Robertson and Bretherton, 1991) and SCATS (Lowrie, 1992; Sims, 1979) is that it is not constrained to generate cyclic signal 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 each phase. These constraints are in the interests of safety and fairness, respectively. ALLONS-D can consider additional priority for vehicles of different types or occupancy levels in the traffic stream. Rolling Horizon, as a strategy for on-line signal control, has been studied by several transportation researchers (Bell and Brookes, 1994). It is a strategy utilized in other tested traffic control schemes such as OPAC (Gartner, 1983a,b), PRODYN (Henry *et al.*, 1983), and UTOPIA (Mauro and DiTaranto,

1990; Mauro, 1991). This paper discusses some of the distinguishing features of ALLONS-D relative to these other methods.

Fixed-cycle signal plans control an intersection by implementing a constant sequence of green, yellow, and red time intervals for each phase. A phase of traffic is defined as the flows of vehicles that may proceed through an intersection without conflict. 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 traffic lights. A procedure to coordinate control for intersections that do not have cyclic signal plans is not as evident. We address this issue in this paper.

We study the degree of progressiveness of the signal plans produced by ALLONS-D. A means to further coordinate these controllers, on an arterial network, to reduce network-wide delay is also described. This coordination scheme can be used to improve travel time performance of transit routes. Several sets of simulations analyzing the usefulness of a passive bus priority system using decentralized local controllers like ALLONS-D will be presented.

This document is organized as follows. A complete description of ALLONS-D is first presented. A survey of other traffic control schemes, in particular those related to ALLONS-D, is provided; this is followed by a comparison of the ALLONS algorithm with dynamic programming approaches. The traffic network simulator that we designed for our simulation experiments of ALLONS-D is briefly described. Extensive simulation experiments on a single intersection were performed. From these tests, comparisons of the performance of ALLONS-D to the performance of fixed-cycle plans are detailed. A discussion of the applicability of active and passive transit priorities is then given. After that, we comment on using ALLONS-D in a two layer signal control architecture, in the interest of network coordination. Finally, some general comments on ALLONS-D, including the issue of computational speed, conclude the paper. We note that a brief description of the main features of the ALLONS-D scheme was given in the conference publication (Porche *et al.*, 1996).

DESCRIPTION OF ALLONS-D

A description of the basic architecture of ALLONS-D is provided at the beginning of this section. We then describe the decision tree used

for optimization and the states in the tree as part of a presentation of the system model. Following that we provide a detailed description of how delay is calculated; finally the search algorithm is explained.

Architecture and Optimization Strategy

The ALLONS-D 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 (typically of the order of 5–15 s), 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.

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 (Michalopoulos, 1991). The intersection controller requires knowledge of the exact time at which the vehicle arrives at the end of the link and the queue that the arrivals join upon 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 perhaps with other information (i.e., probe vehicle data).

Decision Tree

ALLONS-D computes the optimal signal switching sequence at an intersection using a strategy based on the dynamic programming principle; this is done each time the controller takes a "snapshot" of the system, as discussed. 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. It is evident 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 Fig. 1.

Each arc 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 yellow–red time plus the minimum green time ($t_{step} = t_{YR} + t_{MG}$). During a yellow–red 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 Fig. 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 Fig. 1, represent the state at

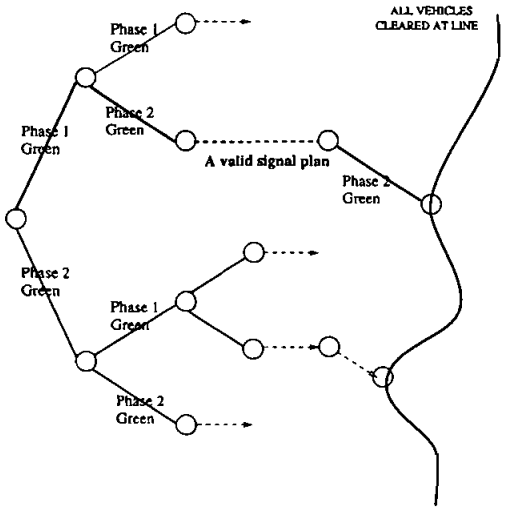

FIGURE 1 Example decision tree.

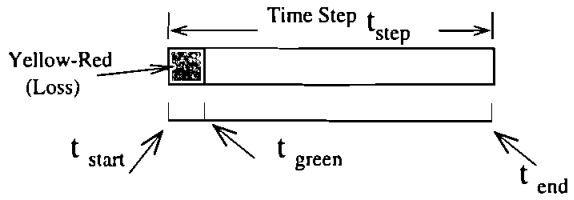

FIGURE 2 Epochs in a time step.

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 ($\bar{q}(t) = [q^1 \cdots q^{\ell_{max}}]'$), (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 size equal to the number of lanes approaching the intersection, denoted ℓ_{max}. The state is again summarized in Eq. (1):

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

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 ($d=0$). Two nodes may be equivalent; if they are 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, then they are the same node and thus the same state. The state of an intersection is updated by extending the duration of the current green phase or changing another phase to green. In the decision tree, 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:

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

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. A bound on the arrival times is determined geographically by the location and distance between neighboring intersections (see Fig. 3).

The update equations for each state component are as follows:

$$x_1[d+1] = x_1[d] + t_{\text{step}}, \tag{3}$$

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

$$x_3[d+1] = x_3[d] + u[d],\ u[d] \text{ is admissible}, \tag{5}$$

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

For example, at an intersection with two phases 1 and 2, if the current phase is 1, then the admissible control is 1 or 0 which corresponds to a decision to switch the signal and a decision to not switch the signal, respectively.

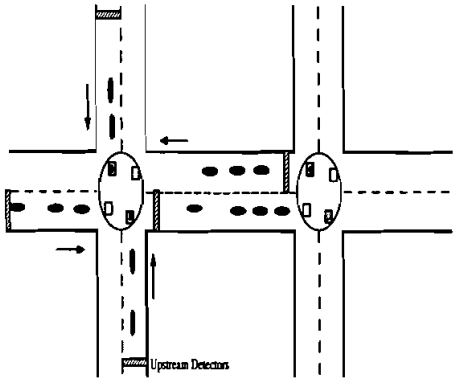

FIGURE 3 A typical intersection snapshot.

For example, if $u(\cdot)=1$ is applied when $x_3[d]=1$, then $x_3[d+1]=2$. The number of departures $D(\cdot)$ (which could be negative if the number of arrivals exceeds the number of departures) is a function of the control variable, the prior queue sizes ($\bar{x}_2$), and the arrivals during the time interval. Note how the parameter t_{step} is updated by the control:

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

Consider the depth-first expansion of nodes along the jth path as shown in Fig. 4. 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 the two states are equivalent. Thus, the graph will 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 decision network), the computational requirements to compare each newly generated node to all existing nodes could be burdensome (this is discussed later in the paper). We chose not to do such comparisons and thus treat the decision network as a tree.

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

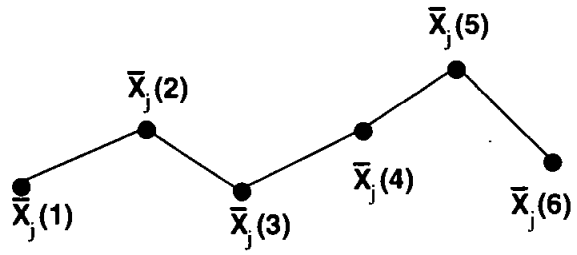

FIGURE 4 A single path.

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.

Delay Calculations

The cumulative delay of all vehicles passing through an intersection is the objective function of the optimizer of ALLONS-D. 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 (Gartner, 1983a). 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 to depart will be referred to as the *headway constraint*. It is calculated using the saturation flow (f_{sat}^{ℓ}) for a link ℓ. The maximum number of vehicles that can leave an intersection is dependent on the saturation flow and the length of the interval of green light ($t_{\text{end}} - t_{\text{start}}$). The saturation flow is link specific, by the symbol $k^{\ell}(\cdot)$.

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.

Delay Equations

We provide below the detailed delay equations used by the ALLONS-D optimization scheme to calculate the arc costs in the decision tree which sum to the delay for a control sequence. The reader is referred to the Appendix for an explanation of the notation.

The *outflow value* ($t^{\ell}_{\text{outflow}}(\cdot)$) reflects the constraints imposed on a vehicle wishing to clear an intersection. It is the absolute time after which a vehicle on a green phase will be able to depart a link. The outflow value observed by vehicle i on link ℓ is shown in Eq. (8):

$$t^{\ell}_{\text{outflow}}(i) = t_{\text{green}} + \sum_{\text{index}=1}^{i-1} h^{\ell}, \quad h^{\ell} = \frac{1}{f^{\ell}_{\text{sat}}}. \tag{8}$$

Four sets of indicator variables are defined that respectively correspond to: (1) a vehicle being *Q*ueued up at the start of the green light; (2) a vehicle being able to *D*epart a link during the green period; (3) a vehicle being partially *C*onstrained by the headway of vehicles in front, and (4) a vehicle being constrained by the headway of vehicles in front of it and *P*rohibited from leaving during the green period:

$$Q^{\ell}(i) = \begin{cases} 1 & a(i) \le t_{\text{green}}, \\ 0 & \text{else}, \end{cases} \qquad D^{\ell}(i) = \begin{cases} 1 & i \le k^{\ell}(t_{\text{step}}), \\ 0 & \text{else}, \end{cases} \tag{9}$$

$$C^{\ell}(i) = \begin{cases} 1 & a(i) \le t^{\ell}_{\text{outflow}}(i), \\ 0 & \text{else}, \end{cases} \qquad P^{\ell}(i) = \begin{cases} 1 & t_{\text{end}} \le t^{\ell}_{\text{outflow}}(i), \\ 0 & \text{else}. \end{cases} \tag{10}$$

Let $p \in \mathcal{P}_I$ represent the phase of the link that vehicle i is on. The delay to vehicle i depends on whether the phase p is green or red. If the phase is red then $Delay_r$ expresses the delay as follows: $Delay_r(i) = [\min(t_{\text{end}} - t_{\text{start}},\ t_{\text{end}} - a(i))] * w(i)$. The delay expression for vehicle i when the phase is green, denoted $Delay_g$, is much more complex. Delay

expressions (11)–(14) below reflect the delay calculation for each relevant vehicle (a vehicle's delay is not considered if it is not observed to arrive during the time step (i.e., $a(i) > t_{\text{end}}$)) to determine the arc cost in the decision tree.

The cases previously enumerated respectively correspond to the following lines of the equation:

$$
\begin{aligned}
&Delay_g(i) \\
&= Q^{\ell}(i) * (1 - D^{\ell}(i)) * \left[(t_{\text{end}} - t_{\text{green}})\right. \\
&\quad \left. + \min(t_{\text{green}} - t_{\text{start}},\ t_{\text{green}} - a(i))\right] * w(i) && (11)\\
&+ Q^{\ell}(i) * D^{\ell}(i) * \left[(i-1) * h^{\ell}\right. \\
&\left. + \min(t_{\text{green}} - t_{\text{start}},\ t_{\text{green}} - a(i))\right] * w(i) && (12)\\
&+ (1 - Q^{\ell}(i)) * (1 - C^{\ell}(i)) * P^{\ell}(i) * [t_{\text{end}} - a(i)] * w(i) && (13)\\
&+ (1 - Q^{\ell}(i)) * C^{\ell}(i) * (1 - P^{\ell}(i)) * \left[(t^{\ell}_{\text{outflow}} - a(i)\right] * w(i). && (14)
\end{aligned}
$$

Notice that each equation contains a weighting factor $w(i)$ for car i. A signal timing plan that prioritizes higher occupancy vehicles should weight the delay to vehicles transporting a larger than average number of travelers. This enables passive priority for higher occupancy vehicles. Hence, a signal plan optimized on delay in this way is *occupant optimal.*

Coupling the States

The decision tree of Fig. 1 highlights how ALLONS-D breaks up the time domain into intervals. These intervals are the arcs of the decision tree. We add a new component to the state, denoted a_{last}. The state component a_{last} serves as a coupling variable between adjacent arcs/time intervals in the decision tree (see Fig. 1) and between the states of consecutive rolling horizon periods (see Fig. 7). Without a_{last}, which is a time stamp of the most recent departure time of a vehicle from the link, certain difficulties would be present and are enumerated as two related problems: (1) the number of vehicles that can leave during a green light is miscalculated; (2) headway constraints from arrivals in one interval of time that effect the next interval will not be accounted for properly.

For modeling purposes, the headway constraint on a link, h^ℓ, as well as the saturation flow, f^ℓ_{sat}, dictate the minimum inter-departure time between consecutive vehicle departures. Because the time-space is discretized, a_{last} is required to account for this constraint. Figures 5 and 6 depict two scenarios. (These scenarios apply to consecutive states in one horizon as well as to consecutive rolling horizon periods.) In each, the determination of the absolute time at which another vehicle may depart, t^ℓ_{outflow}, is different. Specifically, in the first example, the next time at which a departure can occur is set to t_{start} (i.e., $t^\ell_{\text{outflow}} \leftarrow t_{\text{start}}$). In the second example, the next time at which a departure can occur is delayed (i.e., $t^\ell_{\text{outflow}} \leftarrow t^\ell_{\text{outflow}} + (h^\ell - |t_{\text{start}} - a_{\text{last}}|)$).

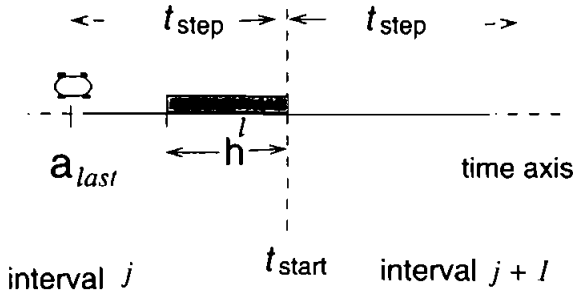

FIGURE 5 An example of when the last departure in a previous time interval does not constrain departures of next time interval.

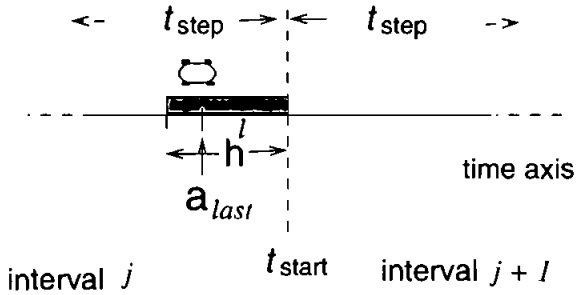

FIGURE 6 An example of when the last departure in previous time interval constrains departures of next time interval.

ALLONS: The Optimization Algorithm

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 essentially uses a branch-and-bound technique and is done in two parts. The first part is a depth-first search to find an initial path in the tree. The second part is a backtracking (Pearl, 1984) procedure. Specifically, the 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. This is equivalent to having all anticipated arriving vehicles on each link cleared from the intersection. The depths of different branches of the tree can thus be different; this is why the boundary of the decision tree is not drawn as a straight line in Fig. 1. As a result, we eliminate the necessity of a terminal cost assignment for boundary states.

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 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 are compared to maintain a current 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. The process is guaranteed to find an optimal solution. In the worst case, the tree could be exhaustively searched. In practice, the pruning from the initial path was shown to be much more efficient. Initially, we performed simple tests which showed that only 10 percent of the state

space was expanded. The "true" demonstration of the feasibility is indicated in the simulation section in which all runs were achieved in "reasonable" real time. The algorithm is outlined below.

1. Begin at the *start node*. Pick a decision (select an initial phase).
2. Choose the next phase, yet to be considered, which incurred the largest cost.
3. Check: If all traffic is cleared then record the policy and its cost and continue; else, go back to 2.
4. Save the value of this policy's cost as the *minimum policy cost* and record the policy as the *minimum policy*.
5. From this node in the path (policy), back up to the parent node.
6. In order of the phases with largest cost, consider all unexplored paths from this node (which are valid):
   (a) Choose the next phase (another node), yet to be considered, which incurred the largest cost.
   (b) Check: If the current path is currently too costly (> existing *minimum policy cost*), then eliminate the node from consideration (prune the tree) and goto 6a.
   (c) Check: If all traffic is cleared then record the policy and its cost, update the values of *minimum policy cost* and *minimum policy* if qualified and continue.
7. If current position in the tree is not the root then goto 5.
8. Return the current record of the *minimum policy*.

STLC-ϵ: A Dynamic Rule

It would be more appropriate to generalize the above STLC concept to one in which a phase change occurs not merely if one phase cost exceeds another but if one phase cost exceeds the others by a specific amount. We define a function $\epsilon(\cdot)$ which can be set to an arbitrary value (i.e., $\epsilon(\cdot) = 0$). It would be practical to define $\epsilon(\cdot)$ as an increasing function of the degree of saturation at the intersection; a very saturated intersection typically requires longer intervals of continuous green time per phase than a less saturated intersection. The criteria for phase changes should be exaggerated for congested intersections relative to uncongested intersections. An example would be to define it as a percentage of the maximum green time t_{MAX} and flow f on the

most congested approach such that

$$\epsilon(f, f_{\text{sat}}) = f/f_{\text{sat}} * t_{\text{MAX}} * \alpha, \quad 0 \leq \alpha \leq 1.$$

No further attempt is made to develop $\epsilon(\cdot)$ in this paper; its use does not affect optimality but could improve the computational efficiency of the search under varying traffic conditions.

Rolling Horizon Implementation

As in OPAC, PRODYN, and UTOPIA 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 (recall that $t_{\text{step}} = \Delta$ when there is no switching). 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, 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 illustrated in Fig. 7.

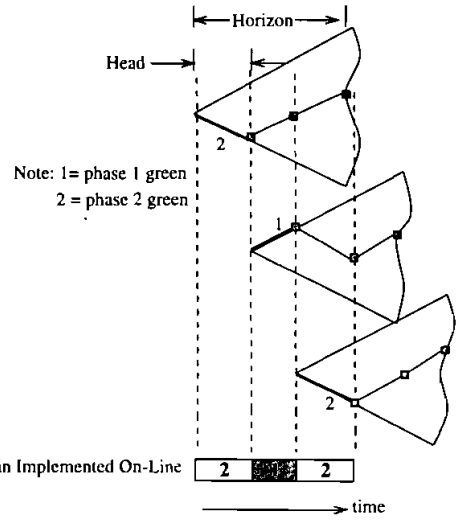

FIGURE 7 Rolling Horizon: Plan results from several rolls.

COMPARING ALLONS-D TO OTHER SCHEMES

There are various traffic responsive systems that determine signalization in real-time based on sensed traffic. Some of these are not constrained to cyclic signal plans. In general, traffic responsive systems can be differentiated along different categories including system architecture, type of signal plans developed, scope of controls and other factors. Numerous systems have been proposed. We refer the reader to (Bell, 1992; Dion and Yagar, 1996) for an overview of these schemes. Only a small subset of these has been either field-tested or made commercially available. This survey will concentrate on these systems.

Adaptive-Cyclic Systems

These systems modify existing parameters like cycle-length, green split, and offset on-line. Updates are usually performed as frequently as 5–10 s in some systems and every 5–10 min in others. The systems SCATS and SCOOT are prime examples. The SCOOT (Split, Cycle and 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 (Robertson and Hunt, 1982). The Sydney Co-ordinated Adaptive Traffic System (SCATS) optimizes cycle length, phase split, and offset so as to maintain the highest degree of saturation in the traffic network. Automatic coordination procedures are usually unique to each system. Both adaptive cyclic schemes SCATS and SCOOT have coordination schemes that involve setting common cycle lengths and determining offsets.

Adaptive-Noncyclic

Systems like UTOPIA (SPOT) (Mauro and DiTaranto, 1990), OPAC (Gartner, 1990), PRODYN (Henry *et al.*, 1983), and SPPORT (Yagar and Han, 1994) attempt to optimize traffic on-line without being confined to a cyclic signal plan – although a cyclic plan is not a prohibited result. The concept is to measure real-time data, determine an optimal (or near optimal) signal plan, and implement the signal policy found – all on-line. Such systems require a prediction or

anticipation of soon-to-arrive traffic in order to adapt the signal policy accordingly. This is a sort of feed-forward control and it requires detectors upstream of the stop line (see Fig. 3). Coordination efforts may or may not be involved. Another proposed scheme similar to ALLONS-D is COP (Controlled Optimization of Phases) (Sen and Head, 1997). It is a decentralized adaptive control scheme for an intersection that assumes that detectors will predict vehicle arrivals sufficiently so that a forward dynamic program can determine an optimal signal plan.

The difficulty in accurately forecasting vehicle arrivals for any lengthy segment (or horizon) of time influences the desire for a rolling horizon implementation of these schemes. Recent studies have shown that short roll periods are sufficient (Bell and Brookes, 1994). How the horizon is chosen and set is a key distinguishing characteristic between various plans of this type. For example, the horizon could be a preset, fixed time period, or it could be variable in some way. Unlike OPAC and UTOPIA, 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.

Another key distinguishing characteristic of a scheme is how an optimal or best suboptimal plan is found (e.g., dynamic programming, heuristic search, expert system, and so forth). In ALLONS-D, the enumeration and search of the decision space, as described earlier in this paper, is more comprehensive than in OPAC since every possible policy is considered. Yet, it is computationally tractable. Furthermore, ALLONS-D attempts more detailed cost calculations as delay is accounted for in its truest definition for each vehicle. 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. There are of course conditions under which the entire decision 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 that we have performed so far suggest that this search technique is computationally implementable in real time.

In UTOPIA and PRODYN, two other rolling horizon 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. Specifically, UTOPIA is a two-level control scheme with a local intersection level control algorithm called SPOT, and an area level controller. The SPOT controller utilizes the rolling horizon implementation strategy with a roll period of 120 s and implementing the first 6 s of calculated signal plans. The area level controller provides weights for the cost functions for each of the SPOT intersection controllers (Nelson *et al.*, 1993). Weights are selected based on macroscopic flow considerations, movement of transit vehicles, data gathered from local controllers, and other considerations. In general, the weights are used to establish preferred routes in the network. 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 that is achieved via the step roll of the horizon every time step.

COMPARISON WITH DYNAMIC PROGRAMMING APPROACHES

Other traffic control schemes like PRODYN (Henry *et al.*, 1983) and COP (Sen and Head, 1997) rely on a forward dynamic programming approach to find a policy that minimizes an approximation of vehicle delay. Thus we are motivated to provide a comparison of the ALLONS-D optimal search procedure to a standard forward dynamic programming approach.

A decision tree of different policies could be represented by a network. The goal is to find the best policy in the network. The best policy is the path with the smallest cost. Clearly, our problem is a shortest path problem in the decision network. Dynamic programming (DP) is an optimization procedure that would seem to be well suited to our application. DP, in general, is applicable to problems requiring a sequence of interrelated decisions (Dreyfus and Law, 1977). Given a network, it is computationally much more efficient than a brute-force enumeration and comparison of all possible paths. Robertson and Bretherton (1974) describe a backwards DP formulation which

determines the optimal signal plan using predictions of queue lengths for a fixed horizon. In comparison, ALLONS-D uses a depth-first branch-and-bound (DFBB) algorithm with STLC branch ordering (which steps forward in time).

As we have described, in order to do an optimal search and perform an exact detailed calculation of delay, the state would have to consist of at least five components (in which two are vector valued). We show by example that there would be little opportunity for aggregation and the burden required to compare nodes would render a standard DP formulation relatively inefficient. Nonetheless, a modified reaching algorithm to solve a forward DP problem is considered. (It is similar to a forward DP strategy proposed by Park *et al.*, 1998.) This modified algorithm, which incorporates the STLC concept and its pruning mechanism, is demonstrated to be less efficient in some examples than the ALLONS algorithm.

Let $\bar{X}[0]$ represent the initial state,

$$V(\bar{X}[\cdot]) = \text{the minimum cost path connecting the initial state } \bar{X}[0] \text{ to } \bar{X}[\cdot]. \tag{15}$$

Clearly, $V(\bar{X}[0]) = 0$ is a boundary condition. Let $C(\bar{X}', u[d])$ represent the cost of a control decision, $u[d]$, to go to state $\bar{X}[d+1]$ from some state, $\bar{X}'$, preceding it. Let $u[d]$ be the control applied to go from the predecessor state $\bar{X}'$ to $\bar{X}[d+1]$. A recurrence relation can be expressed as follows:

$$V(\bar{X}[d+1]) = \min_{\{\bar{X}' \in \{\text{parent nodes of } \bar{X}[d+1]\}\}} [C(\bar{X}', u[d]) + V(\bar{X}'[d])]. \tag{16}$$

Examples: Comparing FDP and ALLONS-D

In Fig. 8, a scenario is presented. This scenario is purposely devised to be one on which the opportunity for node aggregation is likely: The vehicle arrivals are sparse. The optimization parameters used are $\Delta = 5$, $t_{MAX} = 25$, $t_{YR} = 2$, $t_{mg} = 3$. The decision tree expanded which enumerates all possible control policies is shown in Fig. 9(a). There are at least 63 nodes in the tree.

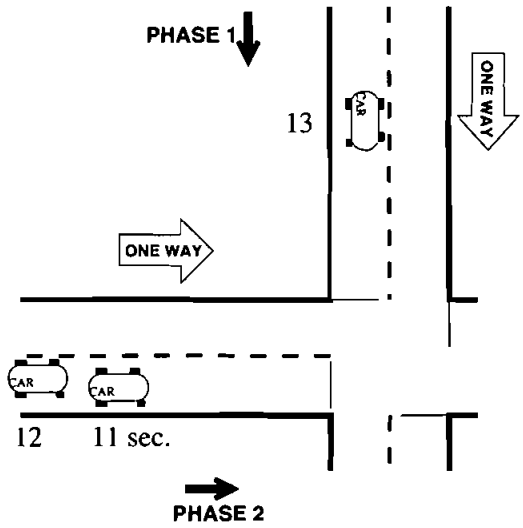

FIGURE 8 Example scenario.

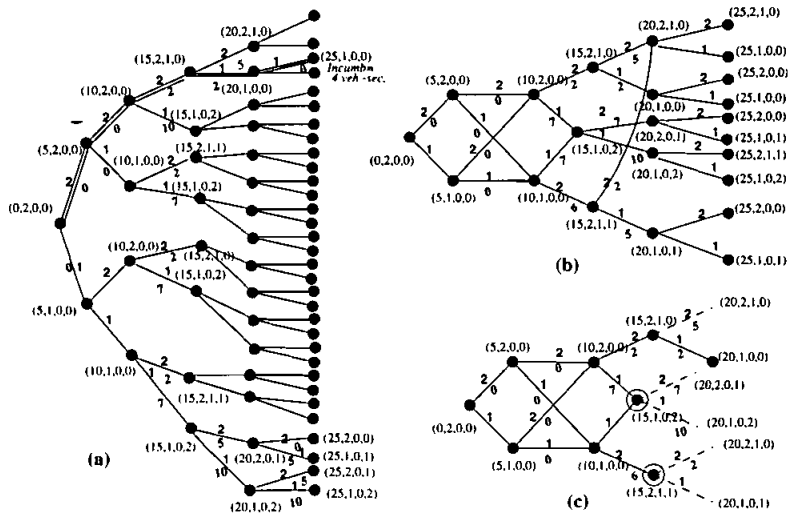

FIGURE 9 (a) Decision tree for the traffic scenario, (b) Resulting DP Network from Standard forward DP, (c) Resulting Network, with pruned nodes encircled. Assume a queue departure rate of 2 cars/5 seconds in the delay calculations.

Consider now a standard DP formulation of the problem. Node aggregation compacts the state space for this example. The result is the DP network shown in Fig. 9(b). There are 20 nodes in this network.

A modified forward DP (FDP) algorithm that incorporates pruning of states is constructed. There are two parts to this DP-based algorithm: (1) a depth first search of an initial policy generated by the STLC rule, and (2) a reaching algorithm to solve an FDP with pruning of states.

The resulting state space with the pruning added to the standard FDP algorithm has only 9 nodes. The result is the decision network shown in Fig. 9(c). In comparison to standard DP formulation, node aggregation alone does not compact the state space as well as the added pruning does for this example. The details of the application of the algorithm for our example is disclosed in Table I.

In comparison, the ALLONS algorithm performs efficiently as well. In Fig. 9(a), the STLC policy is represented by a double line and corresponds the following state transitions:

$$(0,2,0,0) \rightarrow (5,2,0,0) \rightarrow (10,2,0,0) \rightarrow (15,2,1,0) \rightarrow (20,1,0,0).$$

This means that starting with phase 2 as the green phase, the traffic signal remains green for this phase 2 for the first 15 s before a switch to phase one for 5 s. This policy results in a total delay of 4 vehicle-seconds. In this particular case, it is an optimal policy.

In Fig. 10, encircled nodes and dashed arcs show the pruned nodes when applying the ALLONS-D algorithm. Only 17 nodes are visited using the ALLONS-D algorithm on this example. Now, suppose that the initial state for phase is one as opposed to two. A different STLC path is found. This STLC policy is represented by a double line in Fig. 11; it is to remain in phase 1 for the first 15 s and switch to phase 2 for the next 10 s. This policy results in a total delay of 12 vehicle-seconds. It clearly is not the optimal policy. Nonetheless, an efficient search for the optimal is performed. Another optimal path is found by visiting 31 nodes.

For a tree, each visit suggests at least one compare operation and one addition in the tree searching procedure. In a standard FDP algorithm like reaching or a variation of such an FDP algorithm, each

TABLE I Calculations for solution of example scenario with modified FDP algorithm

$\mathcal{T}$	$\mathcal{P}$	*Current state*	*New state*
(0, 2, 0, 0, 0), (5, 2, 0, 0, 0), (10, 2, 0, 0, 0) (15, 2, 1, 0, 2), (20, 1, 0, 0, 4), (25, 1, 0, 0, 4)	∅	(0, 2, 0, 0, 0)	(5, 2, 0, 0, 0)
(5, 1, 0, 0, 0), (5, 2, 0, 0, 0), (10, 2, 0, 0, 0) (15, 2, 1, 0, 2), (20, 1, 0, 0, 4), (25, 1, 0, 0, 4)	(0, 2, 0, 0, 0)	(5, 1, 0, 0, 0)	(10, 2, 0, 0, 0) (10, 1, 0, 0, 0)
(5, 2, 0, 0, 0), (10, 1, 0, 0, 0), (10, 2, 0, 0, 0) (15, 2, 1, 0, 2), (20, 1, 0, 0, 4), (25, 1, 0, 0, 4)	(0, 2, 0, 0, 0) (5, 1, 0, 0, 0)	(5, 2, 0, 0, 0)	(10, 2, 0, 0, 0) (10, 1, 0, 0, 0)
(10, 1, 0, 0, 0), (10, 2, 0, 0, 0), (15, 2, 1, 0, 2) (20, 1, 0, 0, 4), (25, 1, 0, 0, 4) (10, 2, 0, 0, 0), (15, 2, 1, 0, 2), (20, 1, 0, 0, 4) (25, 1, 0, 0, 4)	(0, 2, 0, 0, 0), (5, 2, 0, 0, 0) (5, 1, 0, 0, 0) (0, 2, 0, 0, 0), (5, 2, 0, 0, 0) (5, 1, 0, 0, 0), (10, 1, 0, 0, 0)	(10, 1, 0, 0, 0) (10, 2, 0, 0, 0)	(15, 1, 0, 2, 7) (15, 2, 1, 1, 6) (15, 2, 1, 0, 2) (15, 1, 0, 2, 7)
(15, 2, 1, 0, 2), (20, 1, 0, 0, 4), (25, 1, 0, 0, 4)	(0, 2, 0, 0, 0), (5, 2, 0, 0, 0), (10, 2, 0, 0, 0) (5, 1, 0, 0, 0), (10, 1, 0, 0, 0)	(15, 2, 1, 0, 2)	(20, 1, 0, 0, 4) (20, 2, 1, 0, 7)
(20, 1, 0, 0, 4)	(0, 2, 0, 0, 0), (5, 2, 0, 0, 0), (10, 2, 0, 0, 0) (5, 1, 0, 0, 0), (10, 1, 0, 0, 0), (15, 2, 1, 0, 2)	(25, 1, 0, 0, 4) (25, 2, 0, 0, 4)	
∅	(0, 2, 0, 0, 0), (5, 2, 0, 0, 0), (10, 2, 0, 0, 0) (5, 1, 0, 0, 0), (10, 1, 0, 0, 0), (15, 2, 1, 0, 2) (20, 1, 0, 0, 4), (25, 1, 0, 0, 4), (25, 2, 0, 0, 4)	—	

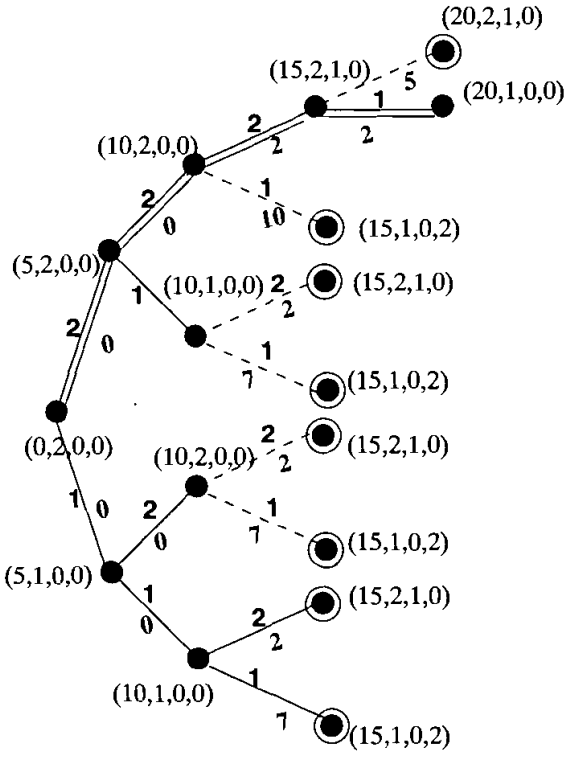

FIGURE 10 Results from ALLONS-D pruning.

node expanded from its parent node in the temporary set $\mathcal{T}$ must be compared to all existing nodes in $\mathcal{T}$ in order to check for aggregation. This is a potential burden in terms of the memory. The algorithm must then allocate memory to store these nodes. By evidence of the example and Table I, there would be a need to efficiently check for previously encountered states, i.e., enable comparisons of newly expanded nodes to members of $\mathcal{T}$. (States encountered could be stored in a hash table in order to check for repeated encounters in constant time; Wellman, 1998.) Additional memory for the set $\mathcal{T}$ would need to be allocated. With the ALLONS algorithm, the expansion of nodes in the decisions tree requires no such considerations and incur no such burdens with ALLONS-D. We explain the computations in Table I that result from applying the modified FDP algorithm. Note that this algorithm is a

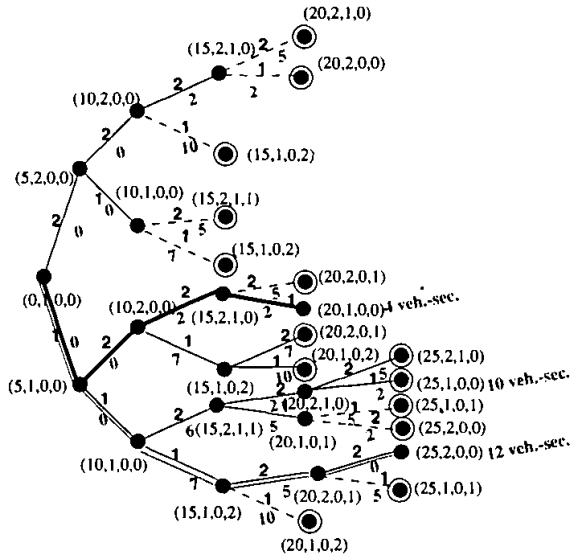

FIGURE 11 Results from ALLONS-D pruning with an alternative initial state.

modification to the reaching algorithm that incorporates pruning of states whenever states are expanded. For the pruning involved, the algorithm uses the cost of the initial STLC policy as the current minimum cost for a policy. Thus, when states are encountered which have a current "cost" that exceeds this value then these are the states that are pruned, i.e., removed from consideration. The states along the STLC path comprise the temporary set $\mathcal{T}$ as the current node, the root node is $(0, 2, 0, 0)$. Initially, the permanent set $\mathcal{P}$ is empty. The new state $(5, 2, 0, 0)$ is chosen for the next iteration. Thus this state becomes the current state added to the temporary set. Simultaneously the state $(0, 2, 0, 0)$ is removed from $\mathcal{T}$ and placed in $\mathcal{P}$. This continues as shown in the table until the set $\mathcal{T}$ is empty.

In conclusion, the traffic control problem generally does not lend itself well to state aggregation. There are sequences of decisions which are almost always undesirable yet would be explored by the standard DP formulation. Certain types of traffic conditions do provide the opportunity to aggregate nodes. These traffic patterns are at the same time inefficiently searched by a decision tree. In the example scenario,

the well spaced arrivals present a state space that would seem to be conducive to node aggregation. Furthermore, we use a sparse state definition a simply (and wrongly) assume that what is denoted is sufficient to calculate delay; this also results in a disregard for minimum and maximum green time constraints. Nonetheless, a great computational advantage (over ALLONS) of a standard FDP algorithm was not observed. A hybrid formulation of the ALLONS search procedure and an FDP was considered. Yet, the (relative) computational simplicity of the ALLONS-D algorithm suggests that even a modified FDP algorithm may not be more efficient.

A SIMPLE TRAFFIC NETWORK SIMULATOR (STNS)

Real-world tests may be impractical for the early development of a new traffic control scheme. It is also impractical to compare different schemes in a real-world setting. Testing via simulation is used here as a practical, albeit less realistic, alternative. For the purposes of this research, we developed an open, uncomplicated and Simple Traffic Network Simulator that we call STNS (Murad and Lafortune, 1995). STNS is a mesoscopic simulator; individual vehicles are accounted for but travel times for a link are determined by an impedance function that considers link occupancy. The impedance function used is similar to a dynamic BPR function. For each link ℓ, it is the following: travel time $= t_0 * (1 + \alpha * (f_\ell / f_{\text{sat}}^\ell)^\beta)$. In this formula, t_0 is the free flow travel time, f_ℓ is the time-averaged flow on link ℓ, and f_{sat}^ℓ is the saturation flow of the link. For simplicity, we use $\alpha = 1$, $\beta = 2$ or $\alpha = 1$, $\beta = 3$ in many of our tests.

Commercially available simulation programs were initially considered. Neither INTEGRATION (Van Aerde *et al.*, 1989) nor TRAFNETSIM (Santiago and Rathi, 1990) can be used to implement and simulate new traffic control schemes – particularly a non-cyclic, real-time traffic control scheme like ALLONS-D. Our STNS simulator has the capability to accept various traffic control modules for signal control, while providing a common testbed for moving vehicles around a user-specified network.

STNS is written in the C programming language; it is currently running in a UNIX operating system environment. It is a time-based

traffic simulator, with a 1 ds time resolution. The user configures a traffic network in a series of input files. These can be used to specify: (1) the topology of the links and intersections; (2) the routes to be used when moving vehicles around the network; (3) the traffic demand on various routes; and (4) a traffic signal control plan for the network of intersections. STNS also provides network simulation statistics at the end of simulation runs. Of these, network travel time, and total network delay at signalized intersections are used for comparison.

SINGLE INTERSECTION SIMULATIONS

The purpose of this section is to describe tests on a single intersection. The tests were designed so that a fixed-cycle signal plan would perform very well. The results show that the ALLONS-D scheme performs comparable and better in most cases for the under-saturated tests. Results from near-saturated and over-saturated conditions suggest the ALLONS-D scheme is clearly more suitable.

The data from a set of simulation experiments are included in this section. A summary of results is shown in Table II with details shown in the Appendix. A single isolated intersection with two conflicting approaches is used. There are two origin–destination pairs in this simple network: W $\rightarrow$ E and N $\rightarrow$ S. The saturation flow rates on the roads are 1200 vph and the links are specified to be a mile long. We compare the delay incurred using signal plans generated by the ALLONS-D scheme with the delay incurred using fixed-cycle plans through a series of tests. The traffic conditions for the different tests range from under-saturated to over-saturated but always with an equal amount of uniform traffic on both approaches. This demand is applied for 20 min, and 40 min of simulation is performed to ensure that all vehicles clear the network. This test is designed such that a fixed plan with equal phase splitting would be the practical signal setting policy. The best fixed-cycle plan for each demand is found by enumerating all possible fixed plans and simulating the use of each to determine the best. An additional test with a time-varying demand (shown in Fig. 12) is also performed (last column of Table II). Certainly for this time-varying case, a fixed-cycle signal plan could not perform as well. For the purpose of further analysis, consider the

TABLE II Simulation results – single intersection network

Demand (vph)	300	400	500	550	600	650	t.v.
No. of cars	198	266	332	368	398	436	195
Best-fixed cycle plan Dly (min)	37.3	58.8	114.2	228.8	502.46	875.7	46.3
ALLONS-D delay (min)	28.5	59.4	114.7	209.4	389.2	732	27.5
Optimization parameters	$\Delta = 5$, $t_{MG} = 10$	$\Delta = 5$, $t_{MG} = 10$	$\Delta = 10$, $t_{MG} = 10$	$\Delta = 10$, $t_{MG} = 10$	$\Delta = 10$, $t_{MG} = 20$	$\Delta = 20$, $t_{MG} = 10$	$\Delta = 5$, $t_{MG} = 10$

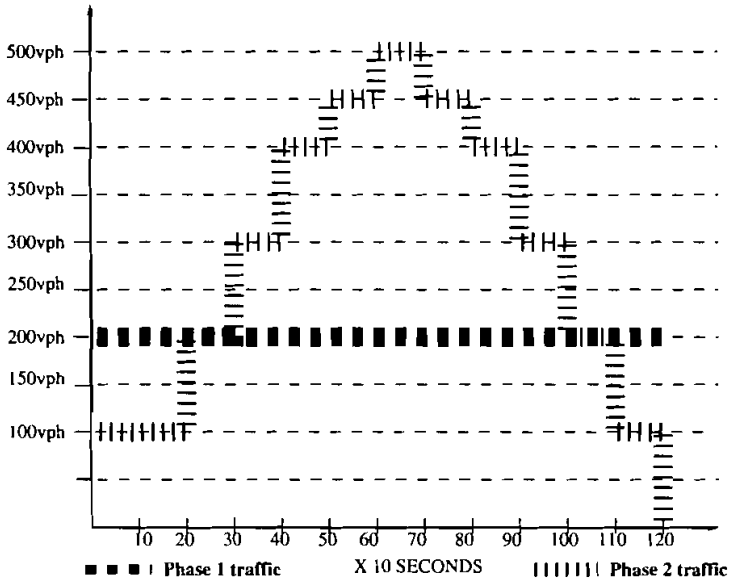

FIGURE 12 Time-varying traffic demand for a single intersection.

"600 vph" test. An examination of the signal plan derived by the ALLONS-D scheme (and shown in Fig. 13) reveals a time-varying "cycle" length. The best fixed-cycle plan for this "saturated" case was far inferior. The average queue size under ALLONS-D control was lower than the average queue size under the best fixed-cycle scheme observed. This is evident in Fig. 14, which compares the time varying queues at the intersection for the best fixed-cycle plan (which is (60, 60)) and the ALLONS-D signal plan.

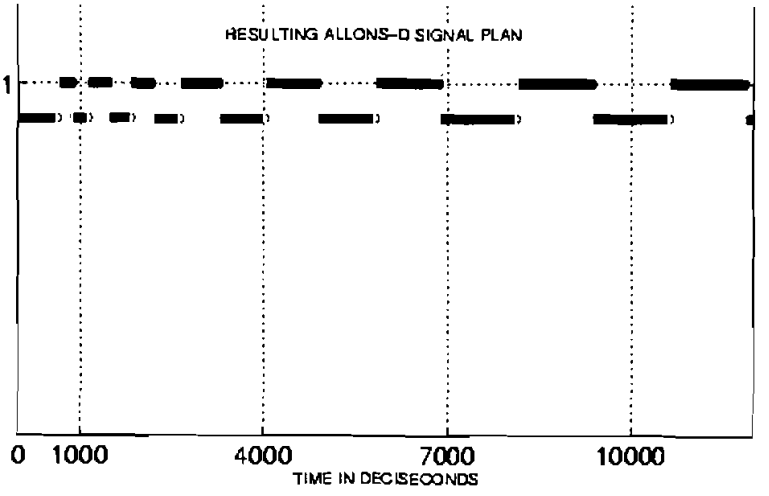

FIGURE 13 600 vph Test signal plan derived by ALLONS-D.

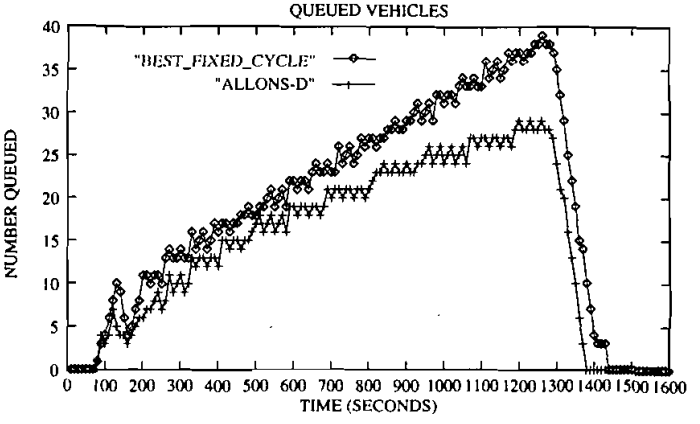

FIGURE 14 600 vph Test: Comparing queued vehicles.

TRANSIT PRIORITIES IN ALLONS-D

Traffic efficiency may be improved by giving priority to public transportation vehicles. Public transportation is ideally a more efficient mode of transportation. It is rationalized that this fact makes busses worthy of priority at signal controlled junctions. This concept is referred to as bus priority or Transit Signal Priority (TSP) (Ins, 1994).

Bus priority systems allow an equipped bus to extend a green light or override a red light. Such rationalization may lead to a proliferation of traffic–transit control systems that allow overrides of a traffic signal. Goals of such systems include allowing a bus that is perhaps behind schedule to save time, reducing overall delay along transit routes, and thus improving consistency on routes. For an overview of some existing systems, the reader is referred to (Nelson *et al.*, 1993; Yagar, 1993). Bus priority systems can be categorized into two types: active priority and passive priority. Active priority systems allow a bus driver (or other transit operator) to command a nearby traffic light directly. The use of active priority may induce additional traffic delay that outweighs any benefit achieved by prioritizing a high occupancy vehicle. Hence, these systems, although being implemented in several cities (Ins, 1994; Ame, 1994), are considered controversial. Passive priority systems seek to weight the importance of a transit vehicle when considering possible signal plans. They are intelligent priority systems that consider the benefits and effects of giving a vehicle priority.

When combined with a responsive traffic control system, TSP has the following goals: (1) decrease the travel time of transit vehicles; (2) minimize traffic delay to regular (background) vehicles; (3) provide a method of contention for priority in the presence of multiple transit vehicles. This section will demonstrate the improved travel time (via reduced delay) to a particular transit line via passive priority. We examine the corresponding system performance as the relative priority of busses is increased in such a manner that active priority is approximated. The effect of loading and unloading of passengers (see Yagar, 1993) is not considered. A small portion of the Ann Arbor traffic network is used for this purpose; it is shown in Fig. 15. Several illustrated examples as well as a simulation experiment (on the network in Fig. 15) are included here to further describe how a decentralized, traffic-responsive signal control system can be made to incorporate priority to high occupancy vehicles (like busses) in an intelligent way.

Simulation Experiment

Two hour-long simulations are performed to demonstrate the use of weighted delay to implement bus priority. The relative weight of transit vehicles (compared to other background vehicles) is varied as a

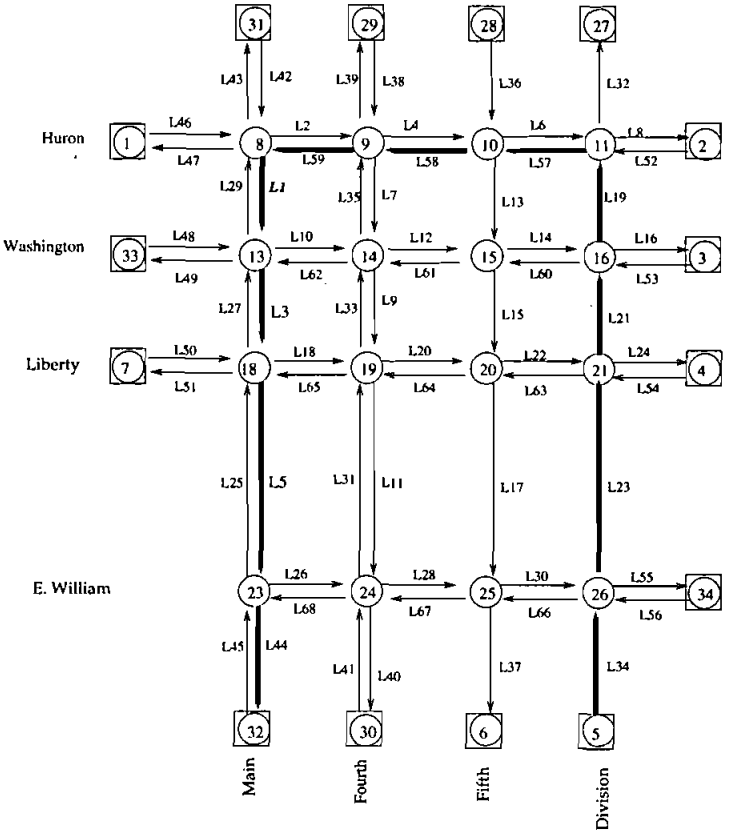

FIGURE 15 Ann Arbor portion.

means to "increase priority". This simulation demonstrates the flexibility of a decentralized scheme to implement either. To a lesser extent, it serves as an analysis of the trade-offs for active vs. passive priority. The higher the relative weight the more likely the effect will be similar to active priority. The first test assigns background traffic, routed down each arterial, to be 200 vph. The transit route is highlighted in Fig. 15. The second test doubles the background traffic to 400 vph. For the first test, delay to transit vehicles along the route is reduced by 30% at the expense of overall travel time and delay increasing by less than 1%. This is seen when the delay to the transit vehicles is weighted four times greater than delay to non-transit

TABLE III Simulation results of 1st test, background traffic 200 vph, transit route 40 vph; 2nd test, background = 400 vph

Relative weight of transit vehicles	*1st test times* (min)			*2nd test times* (min)		
	System delay	*System travel*	*Transit rt. delay*	*System delay*	*System travel*	*Transit rt. delay*
1	1098.7	3015	38.4	2703.2	6642.6	48.8
2	1103.6	3019.7	33.8	2791.4	6731.4	43.3
3	1090.2	3006.4	29.3	2830.7	6769.8	34.5
4	1102	3018.1	27.2	2849.1	6788.3	30.5

vehicles. For the second test, weighting the delay experienced by transit vehicles reduces their delay by 38%. The cost incurred is a 5.4% increase in overall system delay which translates to just over 2% increase in travel time. Table III outlines the results for these two sets of tests. This weighting scheme essentially provides a degree of coordination for the transit route among the intersections that experience the transit demand. This method will be exploited to provide a degree of "coordination" for an arterial network.

Illustrated Examples

Consider the situation shown in Fig. 16. Arrival times of the vehicles are noted in minutes and seconds. The signal plan in the lower right corner is a graphical representation of the result that the ALLONS-D algorithm computes when no additional weighting is given to a bus (i.e., *weight*(bus) = *weight*(car)). From this perspective, the single vehicle on phase 2 should wait and allow the three vehicles on the opposing phase to pass through. In terms of the number of passengers being delayed, this signal plan is sub-optimal. In contrast, the optimization for the scenario in Fig. 17 distinguishes between vehicle types and applies a greater weight to the bus when tabulating cost. The weight for the bus (high occupancy vehicle) is chosen to be five times the weight of the single occupancy vehicle. Appropriately, the plan resulting from the ALLONS-D optimizer allows the bus to preempt the three single occupancy vehicles. Case 3 depicted in Fig. 18 has an additional arriving bus on phase one. Consider how a blind bus priority system would handle this situation (see Fig. 18). The bus on phase two arrives first and could override the signal. This would be

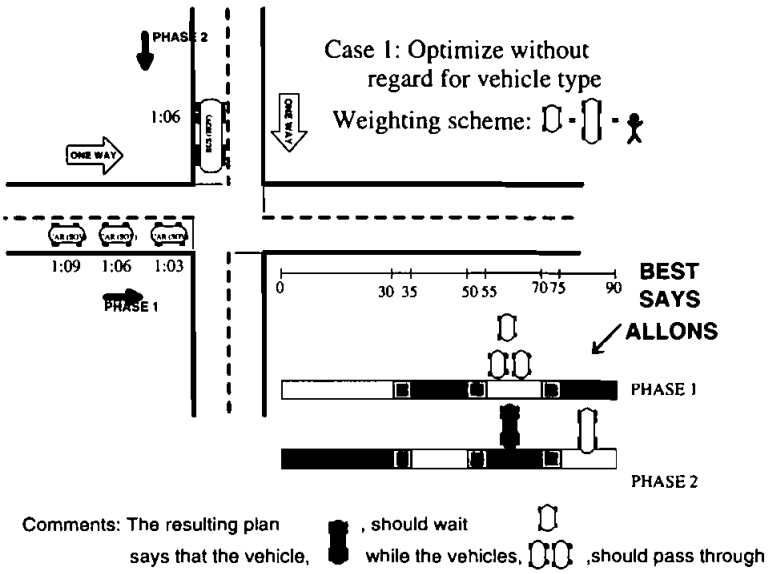

FIGURE 16 Case 1: "control" case – no priority.

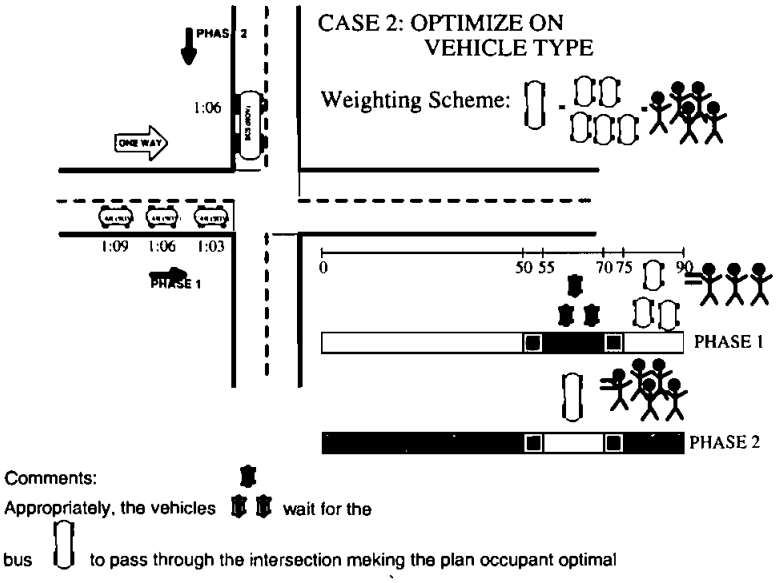

FIGURE 17 Case 2: Priority weighting for busses.

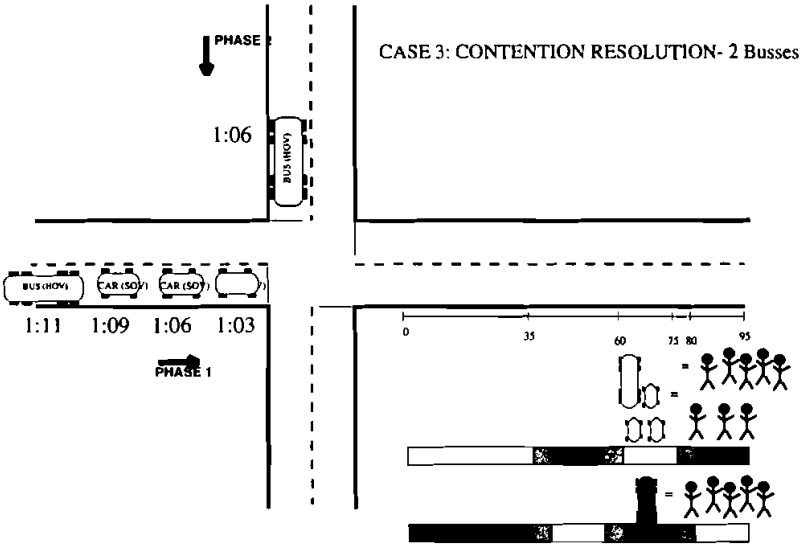

FIGURE 18 Case 3: Contention resolution demonstrated.

sub-optimal as another bus arrives on the opposite phase slightly later but still preceded by several vehicles. The signal plan selected by ALLONS-D correctly chooses the occupant optimal signal plan because it considered the cost of delaying opposing traffic.

ALLONS-D INTERSECTION COORDINATION

There is an implicit coordination among intersections in the ALLONS-D architecture that occurs without receiving explicit coordination commands. This is due to the incorporation of data from upstream, arrival predicting detectors and the rolling horizon implementation. Consider the network and demand shown in Fig. 19. The signal policies resulting from each intersection using ALLONS-D are shown in Fig. 20. Observe how easily it is to extract through-bands. Note that for the signal plans shown in the figure, a green period for

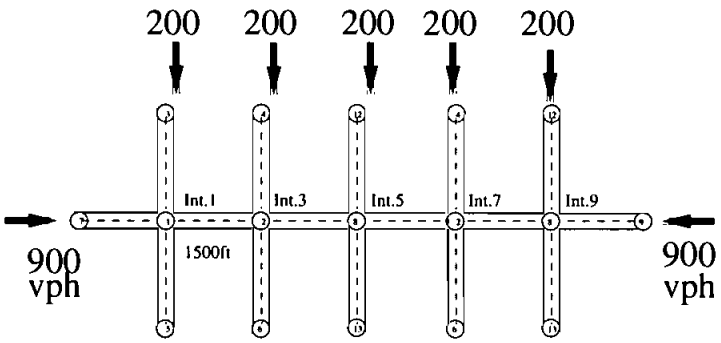

FIGURE 19 Five intersection arterial.

the E–W direction is shown as the lighter, slightly raised portion. The green times at each intersection appear to be offset sufficiently to support reduced stops for platoons traveling down the main street. A signalization scheme that supports green waves and platoons of smooth flowing vehicles is a form of coordination.

For a network of intersections, an optimal signal plan at each local controller does not imply that the network performance is optimal. A means to *explicitly* drive the local optimum towards a system optimum is sought. We propose an explicit coordination scheme for the local, adaptive ALLONS-D controllers in the form of a two-layer hierarchy (see Fig. 21). The local layer of control implements ALLONS-D as described before. The network layer determines coordination requirements. Coordination requirements are defined as directives to favor one phase of traffic over another using weights. These requirements could be time varying. We use the notation $L_i(t)$ to represent a sequence of preferences for intersection i. $L_i(t)$ would indicate for each discrete time period t the relative importance that one phase of traffic has over another. To do this, an intersection affords a higher weight to delay incurred on one phase relative to the other phases at the intersection. The concept is to use $L_i(t)$ to weight the objective function used by ALLONS-D. The process to develop these weights is not discussed here (see Porche, 1998, Chapter 3 for a detailed study of this problem). We show that for an arterial network, system performance is improved when compared to unbiased (i.e., with unity weights for all phases) decentralized controllers.

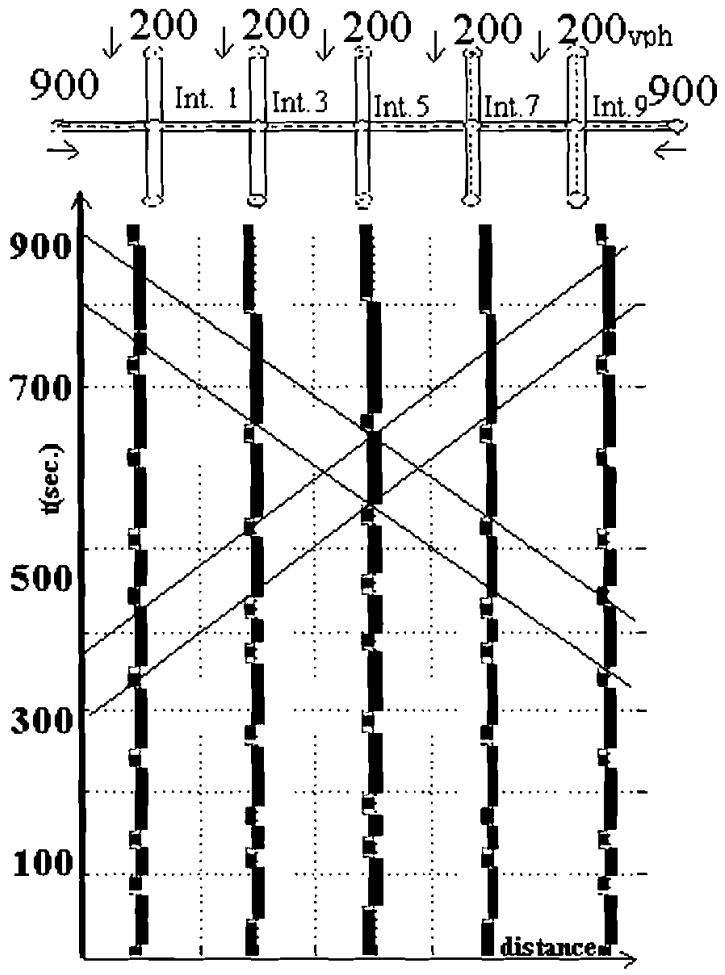

FIGURE 20 Arterial Network and ALLONS-D Signal Plan determined by simulation tests with ALLONS-D.

Simulation Experiments: Comparison Using an Arterial Network

This section describes a number of sets of tests on the three intersection arterial network shown in Fig. 22. Each test assigns the

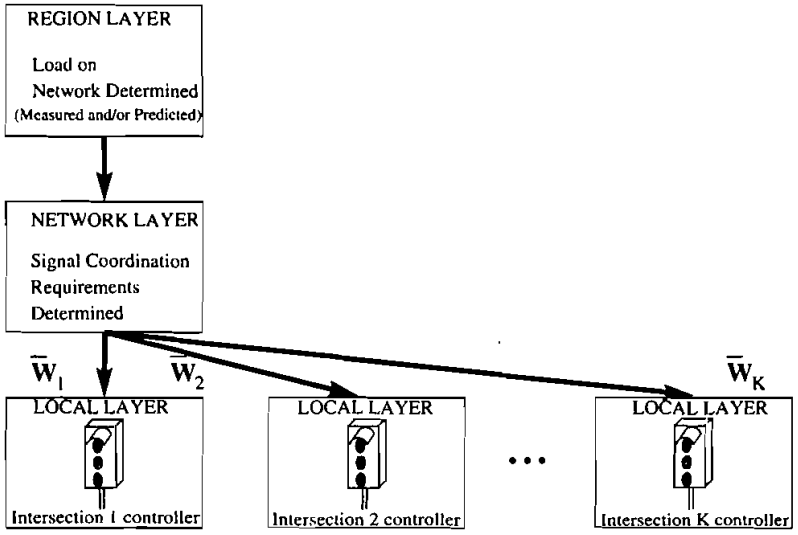

FIGURE 21 Multi-layer signal control architecture.

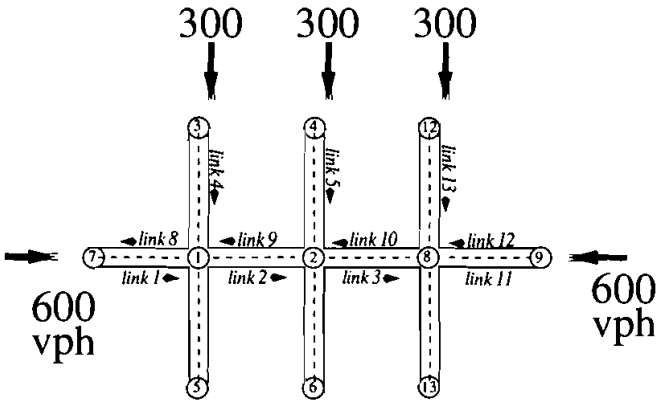

FIGURE 22 Demand on arterial.

same ten minute load on the network but with a varying amount of relative weight on the N/S traffic delay. Each data point on the X-axis of Fig. 23 represents the result of a different test. The results are summarized as follows: (1) there is reason to believe that the

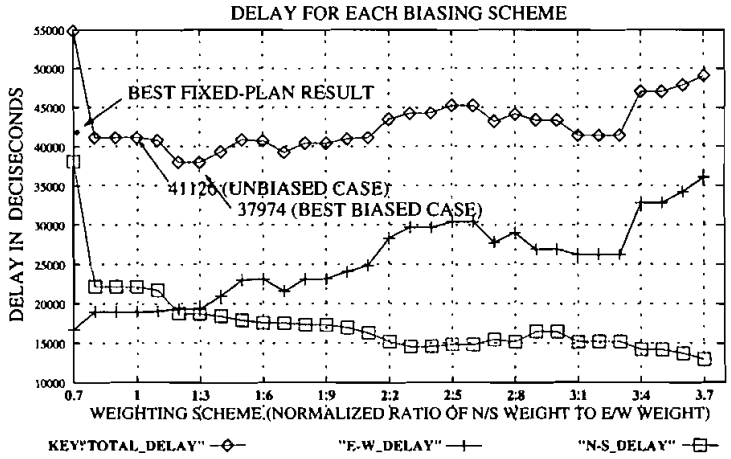

FIGURE 23 Arterial delay results.

TABLE IV Simulation results – summary of search for best fixed cycle plan for demand shown in Fig. 22; units in ds

Total delay (ds)	*W–E delay* (ds)	*N–S delay* (ds)	*Phase splits* (λ_1, λ_2)	*Cycle length* C_l (s)	*Middle intersection offset*	*Comments* $C_l = (1.5*L+5)/(1-Y)$ $g_{eff} = C_l - L$
50478	29358	21120	(0.33, 0.67)	48 ($g_{eff} = 38$)	23	$C_l <$ Webster Eq.
45674	21950	23724	(0.33, 0.67)	44 ($g_{eff} = 34$)	23	$C_l <$ Webster Eq.
41722*	21702	20020	(0.33, 0.67)	40 ($g_{eff} = 30$)	23	$C_l <$ Webster Eq.
43257	23394	19863	(0.33, 0.67)	38 ($g_{eff} = 28$)	23	$C_l <$ Webster Eq.

ALLONS-D control scheme outperforms conventional fixed-cycle methods; and (2) the system-wide performance of the ALLONS-D scheme can be improved by universally providing additional weights to traffic on certain phases in a systematic manner.

First, an exhaustive search for finding the best fixed-cycle plan with appropriate offsets was performed. The standard formula of Webster was used to determine signal plans and suggest a cycle length. An appropriate offset for the middle intersection was determined after extensive simulations. Some of the data is reported in Table IV.

The data in Fig. 23 indicates that the unbiased ALLONS-D controllers are competitive with the coordinated fixed-cycle signal plans since the delay incurred was almost two percent lower (41126 vs 41722 ds). The performance of the unbiased controller was improved by assigning more weight to the N–S traffic on the arterials which resulted in delay that was 9% lower than the best coordinated fixed-cycle signal plans. Analysis of the queue lengths resulting from ALLONS-D control with different weighting schemes was done. It was seen that the unbiased controller, for this specific example, allowed longer queues and excessive delay to occur on the N–S traffic (i.e. the minor road traffic) relative to control with more weight to traffic in that direction. The result was undue delay on the minor approaches as shown in Fig. 23.

Essentially, the use of weights for certain phases at the intersections of a network is utilized as a tool. The purpose is to coordinate the control action of the individual controllers at the intersections in an effort to improve the network-wide performance. As described for a similar problem in (Fernandez and Friesz, 1983), this coincides with a concept from welfare economics: decentralized control to optimize local objectives leads to optimization of global objectives if the externalities are internalized. Along these lines, the weights factor in the global effect of any local decision.

COMPUTATIONAL SPEED AND PARAMETERS

In all of the tests reported in this paper – in which the demand on the network was undersaturated – the measured computation time for the ALLONS algorithm was under 1 s. For oversaturated intersections, in which a signal policy cannot reduce queues and queues grow with time, there is a much higher computational burden on the ALLONS algorithm that is indeterminable in the case where the state space size exceeds the memory capabilities of the device performing the calculation. This is a result of the algorithm's procedure to find a policy that signals through all vehicles queued and those en-route.

For the undersaturated tests described, the lowest value for Δ was 5. The speed of the algorithm is very dependent on the value of Δ. Large values of Δ accelerate the computation time by effectively

reducing the size of the state space. Very small values of Δ (in the experiments described, this corresponds to $\Delta < 3$) also increase the memory storage requirements by inflating the size of the state space. This is true to a lesser extent for the size of other parameters (i.e., maximum and minimum green time).

Fortunately, the need for a small Δ, e.g., $\Delta = 5$ lessens with the degree of saturation. It is generally accepted that a congested intersection desires a policy with a switching frequency that is no less than 5 or 10 s. It is evident in Fig. 13 – in which an intersection is saturated – that ALLONS-D is implementing signal plans in excess of these values. Thus, an adjustment of Δ to 15 or 20 (or higher) can be sufficient to lower the computational speed. In our experiments with saturated intersections, such values for Δ reduced the computation time down to or below a few seconds. This adjustment could be incorporated into ALLONS-D to be a real-time, on-line adjustment. These and other considerations of dynamically adjusting the optimization parameters are left as future work. It should be noted that computational speed measurements were taken while the ALLONS algorithm was running on a Sun SPARC Ultra workstation.

The optimization algorithm of ALLONS, in combination with a rolling horizon implementation, suggests the relevance of the *minimum first arc problem* that is considered by Lark *et al.* (1995). ALLONS is not limited to implementing only the first decision of an optimal policy before rolling. But, if a means to determine the best first decision without finding the entire policy is possible, then a more efficient search algorithm for the implementation used in this paper is possible. The computational speed of the existing algorithm is sufficient. Hence, this approach was not pursued. However, to pursue the approach described in (Lark *et al.*, 1995), the appropriate heuristic function $h(n)$ that would be required is not obvious for the signal control problem described in this paper.

The length of the horizon in ALLONS affects the computational complexity and is determined by the geometry of the intersection and the arrivals into the intersections. A shorter horizon, with all other parameters fixed, reduces computational complexity. Thus, the work of Garcia and Smith (1997) is relevant; it addresses the issue of determining a minimal planning horizon to ensure that the first decision is optimal. In the ALLONS-D scheme, the horizon is finite

and tractable and thus the investigation of a stopping rule as suggested by Garcia and Smith is not considered here.

DISCUSSION: COMPARISON TO A^*

The A^* algorithm initially proposed in (Hart *et al.*, 1975) can be used to solve the shortest path problem in a network. The algorithm relies on obtaining an estimate of the "cost-to-go" ($h(n)$), i.e., a cost to reach the end node from the current node n. The ALLONS algorithm computes the optimal policy. If the heuristic function $h(n)$ can be proved to be conservative (i.e., the cost to go is guaranteed to be less than or equal to the actual cost) then the same can be said concerning the policy found by the A^* algorithm.

A significant point expressed in this work is that any expanded node in the decision tree, for states described in the manner we present, is likely to be unique; the need for the expanded state definition is (i) the detailed calculation of delay and (ii) the incorporation of real-world constraints e.g., maximum green time. Therefore, efficient techniques to find optimal solutions in a network, such as forward dynamic programming and A^*, may not be as useful in the present context.

The use of a conservative "cost-to-go" ($h(n)$) would be a useful modification to the ALLONS algorithm. Formulating a conservative $h(n)$ is not trivial for the traffic signal problem. Nonetheless, we suggest the following procedure called the Interleaved Arrival Procedure (IP):

1. Given data on arrivals to the intersection on multiple right-of-ways (phases), the arrivals are combined into one fictitious queue of vehicles in a lane.
2. Assume a queue departure rate.
3. Duplicate vehicles with identical arrival times are considered as one.
4. $h(n)$ is the cumulative delay incurred upon moving all vehicles from the head of queue at the assumed departure rate until all vehicles have exited the intersection.

We *suggest* that this is a conservative estimate of delay (without any formal proof) since no switching penalties are incurred and it is assumed that cars do not encounter a red light; they are delayed only by the presence of earlier arriving vehicles.

CONCLUSIONS

Our objective in this paper was to describe the main features of the ALLONS-D scheme for real-time control of traffic networks. In particular, we have emphasized those features that distinguish ALLONS-D from other adaptive schemes based on the rolling horizon technique.

A novel branch-and-bound optimization technique is described that finds the delay minimizing signal policy. It could easily be utilized to determine optimal signal policies based on other criteria like the number of stops. The efficiency of the search algorithm was demonstrated through simulation experiments described in the paper.

The ability to incorporate active and passive bus priority using decentralized controllers, like ALLONS-D or other similar schemes, was described and tested in a number of simulation experiments. Furthermore, we explored the concept of providing network-wide optimization for arterial networks using decentralized, adaptive controllers. Our recent work reported in (Porche and Lafortune, 1998) includes the development of general rules for inducing network-wide optimal behavior from decentralized controllers on arterial networks.

We emphasize that the search algorithm employed by ALLONS-D is guaranteed to return an optimal solution for the decision tree and objective function under consideration. We note that this is in contrast with the constrained search approach of OPAC described in the literature. We also note that it would be highly desirable to perform experiments that would compare ALLONS-D with other adaptive-noncyclic schemes and with adaptive-cyclic schemes such as SCATS and SCOOT. This is not possible at this time due to the proprietary nature of some of these systems and due to the lack of a comprehensive software environment to perform such comparisons.

Acknowledgments

The authors wish to acknowledge the thoughtful comments of the anonymous reviewers. It is also a pleasure to acknowledge the important contributions of other researchers in the development of ALLONS-D scheme, most notably Meera Sampath and Raja Sengupta. The initial proposal of utilizing a decision tree for

optimization purposes and the rolling horizon concept was reported in (Chen *et al.*, 1995; Sengupta *et al.*, 1995). These initial concepts were extensively studied on a single intersection using a specially designed simulator called SITA (Single Intersection Test of ALLONS-D). The results from these tests, which were reported in (Porche and Lafortune, 1995), led to a significant redesign of most of the major building blocks of ALLONS-D, including the search procedure, calculation of delay, decision tree state vector, and objective function (to support transit priorities and signal coordination); the main features of the new design were first reported in (Porche and Lafortune, 1995). These reflect the current state of ALLONS-D as summarized in (Porche *et al.*, 1996). Finally, we gratefully acknowledge the contributions of Yawar Murad towards the expansion of SITA into the network simulator STNS (Murad and Lafortune, 1995).

References

Ame. Priority system speeds public transit for Washington city. *American City and County*, p. 47, March 1994.

M.G.H. Bell and D.W. Brookes. The optimisation of traffic signal control over a rolling horizon. *Transportation Systems*, pp. 1013–1018. IFAC, 1994.

M.G.H. Bell. Future directions in traffic signal control. *Transport Research: A*, **26A**(4): 303–313, 1992.

Y.-L. Chen, Stéphane Lafortune, Meera Sampath and Raja Sengupta. A rolling horizon based architecture for traffic signal management. Preliminary report, EECS Dept., June 1995.

F. Dion and Sam Yagar. Real-time control of signalised networks – different approaches for different needs. In *Road Monitoring and Control*. IEE, April 1996.

DOT 1996 Building the ITI: Putting the National Architecture into Action. Technical report, U.S. Department of Transportation Federal Highway Administration, April 1996.

Stewart Dreyfus and Averill Law. *The Art and Theory of Dynamic Programming*. Academic Press, 1977.

J. Enrique Fernandez and Terry L. Friesz. Equilibrium predictions in transportation markets: The state of the art. *Transportation Research*, **17B**(2): 155–172, 1983.

Alfredo Garcia and R.L. Smith Minimal forecast horizons through monotonicity of optimal solutions. Technical Report 97-16, The University of Michigan, 1997. Department of Industrial and Operations Engineering.

Nathan Gartner. Demand-responsive decentralized urban traffic control. Technical report, U.S. Department of Transportation, February 1983.

Nathan Gartner A demand responsive strategy for traffic signal control. *Transportation Research Record*, (906): 75–81, 1983.

N.H. Gartner OPAC. In J.P. Perrin, ed., *Control, Computers, Communications in Transportation: selected papers from the IFAC Symposium*, Paris, France, 19–21 September 1989, number 12 in IFAC Symposia Series, pp. 241–244. Pergamon Press, 1990.

P.E. Hart, N.J. Nilsson and B. Raphael A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, **4**(2): 100–107, 1975.

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, September 1983.

Ins 1994 Transit agencies, cities in Seattle area seek AVI for signal priority. *Inside IVHS*, pp. 10–14, September 1994.

James, W. Lark, Chelsea C. White and Kirsten Syverson A best-first search algorithm guided by a set-values heuristic. *IEEE Transactions on Systems, Man, and Cybernetics*, **25**: 1097–1101, July 1995.

P.R. Lowrie. *SCATS: A Traffic Responsive Method of Controlling Urban Traffic Control*. Roads and Traffic Authority, 1992.

V. Mauro and C.D. DiTaranto. UTOPIA. In J.P. Perrin, edi., *Control, Computers, Communications in Transportation: selected papers from the IFAC Symposium*, Paris, France, 19–21 September 1989, number 12 in IFAC Symposia Series, pp. 245–252. Pergamon Press, 1990.

V. Mauro. Road network control. In M. Papageorgious, ed., *Concise Encyclopedia of Traffic and Transportation Systems*, Advances in systems, control, and information engineering, pp. 361–366. Pergamon Press, 1991.

P.G. Michalopoulos. Vehicle detection video through image processing: The autoscope system. *IEEE Transactions on Vehicular Technology*, **40**(1): 21–29, February 1991.

Y. Murad and S. Lafortune. STNS: a Simple Traffic Network Simulator. Technical Report CGR-95-13, University of Michigan, 1995. Technical Report.

J.D. Nelson, D.W. Brookes and M.G.H. Bell. Approaches to the provision of priority for public transport at traffic signals: a European perspective. *Traffic Engineering and Control*, pp. 426–431, 1993.

Yunsun Park, Isaac Porche, Robert L. Smith and Stéphane Lafortune. An efficient dynamic programming algorithm for signaling, 1998.

Judea Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

I. Porche and S. Lafortune. Dynamic traffic control: Decentralized and coordinated methods using a limited lookahead strategy. Technical Report CGR-95-12, College of Engineering Control Group, the University of Michigan, September 1995. Revised Dec. 1995.

I. Porche and S. Lafortune Coordination of local adaptive traffic signal controllers. *American Control Conference*, June 1998. Philadelphia, PA.

I. Porche, Meera Sampath, Y.-L. Chen, Raja Sengupta and Stéphane Lafortune. A decentralized scheme for real-time optimization of traffic signals. In *Proceedings of the 1996 IEEE International Conference on Control Applications*, pp. 582–589. IEEE, September 1996.

Isaac R. Porche. Dynamic Traffic Control: Decentralized and Coordinated Methods. PhD thesis, The University of Michigan, Ann Arbor, 1998.

Dennis Robertson and Robert Bretherton. Optimum control of an intersection for any known sequence of vehicle arrivals. In *Traffic Control and Transportation Systems*, pp. 3–17. IFAC, September 1974.

D.I. Robertson and R. David Bretherton Optimizing networks of traffic signals in real time – the SCOOT method. *IEEE Transactions on Vehicular Technology*, pp. 11–15, February 1991.

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

Alberto J. Santiago and Ajay K. Rathi. Urban network traffic simulation: TRAFNETSIM program. *Journal of Transportation Engineering*, **116**: 734–744, Nov/Dec 1990.

Suvrajeet Sen and K.L. Head. Controlled optimization of phases (COP) at an intersection. *Transportation Science*, (31): 5–17, 1997.

Raja Sengupta, Meera Sampath, Y.-L. Chen and Stéphane Lafortune. ALLONS: Adaptive limited lookahead optimization of network signals. Technical Report CGR-95-10, University of Michigan, December 1995.

Arthur G. Sims. SCATS: the Sydney co-ordinated adaptive traffic system. *Proceedings of the Engineering Foundation Conference on Research Priorities in Computer Control of Urban Traffic Systems*, pp. 12–27, 1979.

M. Van Aerde, J. Voss and G. McKinnon. *INTEGRATION Simulation Model User's Guide*. Queen's University, 1989.

Michael P. Wellman. Personal communication with Michael Wellman on May 13, 1998.

Sam Yagar and Bin Han. A procedure for real-time signal control that considers transit interference and priority. *Transportation Research-B*, **28B**: 315–331, August 1994.

Sam Yagar. Efficient transit priority at intersections. *Transportation Research Record*, (1390): 10–15, 1993.

APPENDIX

Summary of Notation

The following is a summary of the symbols used in this paper.

$a(i)$	the time at which vehicle i arrives at the intersection ready to depart
a_{last}	the time of the last observed vehicle departure as a result of the most recent control decision to maintain a phase as green. It is a coupling parameter between intervals in a green period
C_ℓ	cycle length
ℓ	a superscript that represents a unique link at an intersection
f_{sat}^{ℓ}	the saturation flow on link ℓ
g_{eff}	Effective green time in a cycle
h^{ℓ}	the headway constraint on link ℓ, also the minimum headway between vehicles on ℓ
i	an index for the vehicles on a link that represents the relative position of the vehicles on the link result of the most recent control between intervals in a green period
$k^{\ell}(t_{\text{step}})$	the maximum number of departures for a time step on link ℓ

λ_j	Phase split for phase j
$\ell_{\max}$	The total number of approaches to an intersection
$\mathcal{P}_I$	a set of integers which correspond to the phases at intersection I
$t^{\ell}_{\text{outflow}}$	the absolute time after which a vehicle can leave link ℓ. It accounts for headway constraints and is updated after each vehicle departure on link ℓ during a green period
t^{ℓ}_{step}	length of time for a single control decision ($t^{\ell}_{\text{step}} = t^{\ell}_{\text{end}} - t^{\ell}_{\text{start}}$)
t_{start}	a reference for the absolute time a control decision is started
t_{green}	a reference for the absolute time a green period begins in a control decision
t_{end}	a reference for the absolute time a control decision ends
t_{MAX}	a constant that represents the maximum continuous time a phase can be green
t_{MG}	a constant that represents the minimum continuous time a phase has to be green
t_{YR}	a constant that represents lost time (which includes the length of the yellow light)
$w(i)$	an optional weight applied to the delay contributed by vehicle i

Details of Single Intersection Tests

Detailed results from an exhaustive enumeration of fixed-cycle plans for the intersection considered in our single intersection simulation are provided here. Table shows delay times in ds.

Signal plan (g_{phase1}, g_{phase2})	300 (vph)	400 (vph)	500 (vph)	550 (vph)	600 (vph)	650 (vph)	*Time varying* (Fig. 12)
(10,10)	22380*	176394	759404	1144475	—	—	58795
(15,15)	23875	35250*	245002	498475	—	—	27785*
(20,20)	26205	39334	320223	605870	897700	1290125	28779
(25,25)	28365	45737	135344	360695	578700	908535	30260
(30,30)	32135	48250	68819*	219740	413340	693375	33935
(35,35)	36380	53915	96298	311131	533171	834428	35079
(40,40)	39960	60026	82159	212425	403650	665100	39223
(45,45)	41600	63583	88537	149178	328610	587426	42813
(50,50)	45355	66914	95477	216313	420145	681168	46540
(55,55)	43575	74927	103761	169610	339240	600040	50439
(60,60)	51400	77855	111875	137324*	301476*	524642	51948
(65,65)	53820	82020	116613	168751	349405	612222	56755
(70,70)	59445	89557	125084	148252	323850	577350	59131
(75,75)	61750	92030	130048	157180	305440	554460	61803
(80,80)	64605	97352	136815	164611	355445	612180	66198
(85,85)	66645	104380	143232	171550	316080	567095	69144
(90,90)	71595	106906	151767	182471	308284	528416	69390
(95,95)	75495	112136	156087	186304	338450	576145	79257
(100,100)	77835	116286	163389	194671	320520	553730	79593
(105,105)	80985	120239	169069	200988	308928	525420*	79593
(110,110)	82450	122148	174816	209953	341490	580498	84523
ALLONS-D	17100	35672	68533	125651	233532	439195	16481