UC Irvine
ICS Technical Reports

Title
Simulation results for traffic signal control

Permalink
https://escholarship.org/uc/item/618752jm

Authors
Leung, Vitus J.
Irani, Sandy

Publication Date
1997

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

SL BAR
Z
699
C3
no. 97-24

Notice: This Material
may be protected
by Copyright Law
(Title 17 U.S.C.)

Simulation Results for Traffic Signal Control

Vitus Leung* Sandy Irani†

Technical Report UCI-ICS-97-24

Department of Information and Computer Science

University of California, Irvine

Irvine, California 92697-3425

May 1997

Abstract

In this paper, we discuss simulation results for the traffic signal control problem. Our algorithms are motivated by theoretical results from

*Supported in part by Caltrans Contract 65Q090. Email: vitus@ics.uci.edu .

†Supported in part by NSF Grant CCR-9309456. Email: irani@ics.uci.edu .

a model for scheduling jobs that may be competing for mutually exclusive resources. The conflicts between jobs are modeled by a *conflict graph*, so that the set of all concurrently running jobs must form an independent set in the graph. We focus on the problem of minimizing the maximum response time of any job that enters the system. For the specific graph which arises in the traffic intersection control problem, we have shown [14] a simple algorithm which achieves the optimal competitive ratio. We have also studied scheduling with conflicts under probabilistic assumptions about the input. Each node i has a value p_i such that a job arrives at node i in any given time unit with probability p_i. Arrivals at different nodes and during different time periods are independent. Under reasonable assumptions on the input sequence, if the conflict graph is a perfect graph, we have given [15] an algorithm whose competitive ratio converges to 1. Using the methodology of Recker, Ramanathan, Yu, and McNally and a modification of their software, we show that some of our algorithms achieve significant improvements over full-actuated control, the most advanced traffic signal control method in the public domain.

1 Introduction

Today's traffic intersection controllers are based on thirty year old signal phasing strategies ([5, 6, 7, 8, 9, 10, 11, 19, 21, 22, 24, 25, 26, 27, 28]). Signal phasings are optimized offline with historical data, downloaded into the controller and triggered by the presence of vehicles. Even the state of the art in adaptive traffic signal control only extend the optimization to a few seconds before every phase change. However, one expected consequence of an effective advanced traveler information system (ATIS) [2, 3, 4, 17] is the rerouting of congested traffic to streets and arterials that may either temporarily be under-utilized or which normally operate below capacity. Under such conditions, signal settings which have been determined based on recurrent traffic demand will not, in general, be "tuned" to accommodate the transient demand generated by the real-time driver information. As a result, system performance (as well as the effectiveness of the ATIS) is limited by the capacity of the signal system to adapt to transient traffic demand. Better strategies will be necessary for many of the proposed Intelligent Transportation Systems.

We consider scheduling jobs which are competing for limited resources. Jobs arrive in the system through time and require a certain set of resources

to be completed. Any two jobs which require the same resource can not be executed simultaneously. We model the conflicts between jobs by a *conflict graph* where each node in the graph represents a type of job. Jobs of the same type have the same requirements. If two types of jobs demand a common resource, there is an edge between those nodes in the graph. Thus, at all times, the set of jobs currently being executed must belong to nodes which form an independent set in the graph. Note that if there are two jobs of the same type in the system, one must wait until the other is completed.

A traffic intersection is depicted in Figure 1. As all drivers know, the traffic on 1 is typically not allowed to proceed with the traffic on 2, 3, 4, 7, or 8. The complete conflict graph for the traffic intersection is also depicted in Figure 1. The intersection controller must schedule the vehicles through the intersection so as to avoid any conflicts. We consider a 'job' to be a platoon or closely spaced line of cars which must pass through the intersection.

It is reasonable to assume that each job requires roughly the same amount of time to execute. Thus, we adopt a discrete model of time and assume that each job requires one time unit to be completed once it is started. At the beginning of a time unit, jobs may arrive on any subset of the nodes in G;

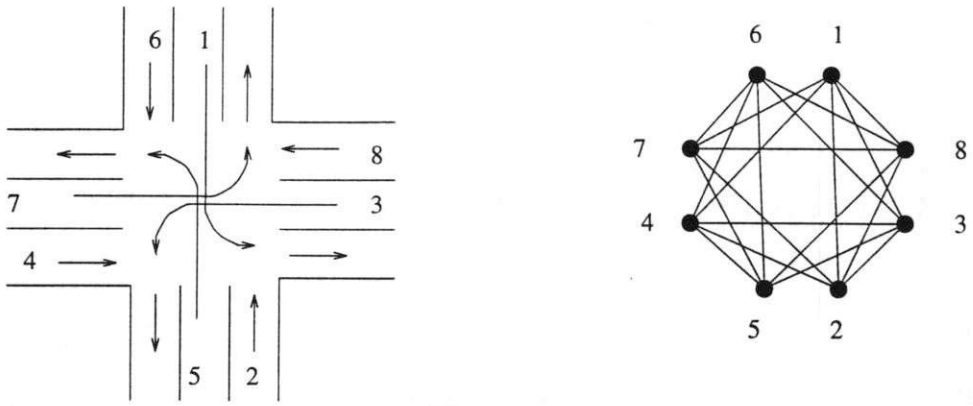

Figure 1: The graph depicting a traffic intersection.

several jobs can arrive on the same node. The algorithm then chooses any independent set of nodes from which to schedule a job for each node. At the end of the time unit, the scheduled jobs are gone from the graph, and all other jobs remain on their respective nodes. Then at the beginning of the next time unit, another set of jobs may arrive.

There are two natural optimization problems that arise from this model. The first is to minimize the *total response time* of all jobs in the system. The second is to minimize the *maximum response time* of any job which enters the system. We focus on bounding the maximum response time of any job which is the maximum, over all jobs j, of $d_j - a_j$, where d_j is the time when j departs and a_j is the time when j arrives. It is important to guarantee the

best turnaround time to any job entering the system.

1.1 Our Results.

Typical scheduling algorithms are faced with the problem of making decisions about which jobs to perform at a given time without any knowledge of the demands that will be made on the system in the future. Such algorithms are said to be *online*. We compare the performance of the online algorithm to the performance of the optimal offline algorithm. An algorithm is said to be *c-competitive* if for any sequence of jobs, its cost on the sequence is at most c times the cost of the optimal offline algorithm on that sequence plus an additive term which is independent of the sequence. The *competitive ratio* of the algorithm is the infimum over all c such that the algorithm is c-competitive.

We argued in [12] and section 1.3 that the problem of scheduling with the traffic intersection conflict graph depicted in Figure 1 is equivalent to scheduling on a $K_{2,2}$. In [14], we described a simple algorithm and prove that it obtains a competitive ratio of 2 on a $K_{2,2}$.

In the upper bound, we devised an algorithm which maintains a set of invariants which bound the accumulation of jobs on edges in the graph. The

lower bound given in [14] shows that the invariants maintained by the algorithm are tight to within a constant factor. For a $K_{2,2}$, we described an adversary which for any positive integer L and any algorithm, can devise a sequence which forces the algorithm to accumulate L jobs on an edge while the adversary has no jobs remaining in the graph. Furthermore, each job in the sequence can be completed by the optimal algorithm within L time units of its arrival. This bound gives a ratio of at least 2 for the competitiveness of any deterministic algorithm on a $K_{2,2}$ where the cost is the maximum response time of a job. Therefore, the simple algorithm described in [14] achieves the optimal competitive ratio. The algorithm picks the side of the graph with the oldest job.

Theorem 1 *The algorithm never has a job older than* $2L$.

We also considered probabilistic assumptions over input sequences in [13]. The class of distributions that we consider can be defined by a vector $D = (p_1, p_2, \ldots, p_n)$. At each time unit, a job arrives on node i with probability p_i. Arrivals in successive time units and on different nodes are completely independent. Any such vector P induces a distribution over arrival sequences which we will call $\mathcal{D}(P)$. For a given algorithm A, we will be interested in

finding an algorithm which minimizes $E_{\sigma \in \mathcal{D}(P)}[\texttt{cost}_A(\sigma)]$, where $\texttt{cost}_A(\sigma)$ is the maximum response time of any job when algorithm A schedules input sequence σ. We are also interested in determining how good our algorithm is in comparison to the optimal algorithm (which knows σ in advance). In the style of analyzing online algorithms against a *diffuse adversary* as defined in [18], we determine

$$\frac{E_{\sigma \in \mathcal{D}(P)}[\texttt{cost}_A(\sigma)]}{E_{\sigma \in \mathcal{D}(P)}[\texttt{cost}_{OPT}(\sigma)]}.$$

We are able to obtain the same bound (to within additive lower order terms) for

$$E_{\sigma \in \mathcal{D}(P)}\left[\frac{\texttt{cost}_A(\sigma)}{\texttt{cost}_{OPT}(\sigma)}\right].$$

It is reasonable to restrict the set of distributions to stable distributions where it is possible to schedule jobs in such a way that the number of jobs in the system returns to 0 with probability 1. Let G^l denote the *extended graph* induced by the job arrivals in the first l time units. This graph is obtained by replacing each node in the graph by a clique whose size is the number of jobs at that node. If two nodes are adjacent in the conflict graph, then the two corresponding cliques are completely connected. The chromatic number of G^l is the number of time units necessary to schedule the set of jobs arriving in

the first l time units. Let $\epsilon_l = l - E[\chi(G^l)]$. Certainly if $\epsilon_l < 0$ for all l, then even the optimal algorithm will accumulate a continually growing backlog of jobs. Thus, the only distributions of interest are those where there exists an l such that $\epsilon_l > 0$.

In [15], the algorithm that we analyze is the simple algorithm which for a given l, gathers all the jobs that arrive in each block of l consecutive time units and optimally schedules them before any of the jobs in the next block of l time units. Note that we are assuming that the conflict graph is small or simple enough that it is feasible to color any induced extended graph either in real time or with some pre-processing. Let G_i^l be the extended subgraph induced by the jobs that arrive in the i^{th} block of l consecutive time units.

Algorithm l-block: Optimally schedule the jobs from the i^{th} l-block starting at the first time unit after the i^{th} l-block finishes and after all jobs from the $i-1^{st}$ l-block have been scheduled.

It is reasonable to think that by combining jobs from consecutive l-blocks (i.e. considering larger l), some improvement in the performance can be gained. It is possible that $E[\chi(G_1^2)] < E[\chi(G_1^1)] + E[\chi(G_2^1)]$, in which case, the 2-block algorithm will perform better than the 1-block algorithm. If this is not the

case, then even the adversary can not combine jobs from consecutive blocks which should lead to a stronger lower bound on the optimal cost. We are able to formalize this intuition in the case where the conflict graph is a perfect graph. In this case, we can prove that as l grows, the competitive ratio of l-block converges to 1. That is, the performance of l-block converges to the optimal offline algorithm. Note that the class of perfect graphs includes both of the applications mentioned earlier.

Theorem 2 *Let G be a conflict graph which is a perfect graph and let P be the distribution vector. Suppose that $\epsilon_1 > 0$. Then there is an algorithm A such that*

$$\frac{E[\texttt{cost}_A]}{E[\texttt{cost}_{OPT}]} = 1 + o(1),$$

where the $o(1)$ is a function which tends towards 0 as the length of the sequence grows.

The algorithm A uses the l-block algorithm, periodically increasing l.

We simulate our simple algorithms using the methodology of Recker, Ramanathan, Yu, and McNally [23] and a modification of their software. Some of our simple algorithms achieve significant improvements over full-actuated control. Full-actuated control is the most advanced traffic signal control method

in the public domain.

1.2 Previous Work.

Minimizing the maximum response time for general conflict graphs is NP-hard: even when all jobs arrive in a single time unit, the problem is equivalent to graph coloring [16]. Even approximating the minimum maximum response time to within a fixed polynomial factor is NP-complete [20]. In [14], we focus on the more traditional worst-case analysis. Our results focused on two special classes of graphs motivated by our applications: interval graphs and bipartite graphs. Such graphs can be optimally colored in polynomial time.

We argued that the problem of scheduling with the traffic intersection conflict graph depicted in Figure 1 is equivalent to scheduling on a $K_{2,2}$. We described a simple algorithm and proved that it obtains an optimal competitive ratio of 2 on a $K_{2,2}$. This result was then generalized for arbitrary bipartite and interval graphs. Although the algorithms for bipartite and intervals graphs are quite different, the bounds they achieve are the same: we proved that for any sequence of jobs, the algorithm can complete every job in time $O(n^3L^2)$, where L is the maximum response time over all jobs in the optimal schedule and n

is the number of nodes in the conflict graph. Note that to achieve a bound on the competitive ratio, we would have had to upper bound the cost of the algorithm by a function which is linear in L.

We obtained a lower bound of $\Omega(n)$ on the competitive ratio of any algorithm on an n-node path. Since a path is both bipartite and an interval graph, this gives a lower bound for both classes. As a result, in [15], we turn to scheduling with conflicts under probabilistic assumptions about the input. Each node i has a value p_i such that a job arrives at node i in any given time unit with probability p_i. Arrivals at different nodes and during different time periods are independent. Under reasonable assumptions on the input sequence, we are able to obtain a bounded competitive ratio for an arbitrary conflict graph. In addition, if the conflict graph is a perfect graph, we give an algorithm whose competitive ratio converges to 1.

1.3 The Traffic Intersection Graph

Notice that in the traffic intersection graph shown in Figure 1 since 1 and 2 are adjacent and have exactly the same neighborhood, they can be merged into one node. The same for 3 and 4, 5 and 6, 7 and 8. The resulting conflict

graph is a $K_{2,2}$. Name the nodes of the $K_{2,2}$ r_1, r_2 on the right side and l_1, l_2 on the left side.

2 Simulation Results

The results presented in this chapter are based upon the methodology of Recker, Ramanathan, Yu, and McNally [23] and a modification of their software. These modifications are described in section 2.1. We present the results for what we call the greedy control algorithm, the block control algorithm, and the greedy block control algorithm in sections 2.2, 2.3, and 2.4, respectively. To evaluate their potential, these control algorithms were applied to the simplified network of five intersections depicted in Figure 2. The network consists of one central intersection surrounded by four intersections operated under traditional full-actuated control (FAC) as described below. In the tests, the operation of the controller managing the central intersection was governed by the respective control algorithms. The results of these tests were then compared to a base case in which the central intersection also followed the traditional full actuated control.

In traditional full-actuated control, the controller steps through a cycle

of prescribed signal phases base upon traffic arrivals and minimum/maximum signal times determined by an offline optimization of previously collected traffic data. Figure 3 shows the standard 8-phase dual ring controller with NEMA (National Equipment Manufacturers Association) signal phases. Even though it is called the 8-phase dual ring, there are only a maximum of 6 phases available in each cycle. We use the same controller and traffic parameters as Recker *et al.*

2.1 Actuated Controller Simulator

The operation of the signals was modified for each of the control algorithms. Additionally, as part of the port of the software to SunOS 5.4, a different random number generator, rand(3F) [1], was used.

2.2 Greedy Control Algorithm

The greedy control algorithm is based on the simple $K_{2,2}$ algorithm described and analyzed in [14]. We used the same minimum and maximum signal times as the full-actuated control to determine the platoon size. Thus, the greedy control algorithm can be thought of as a full-actuated control in which any

NEMA phase can follow any other NEMA phase, with the next phase being selected by the simple $K_{2,2}$ algorithm.

Table 1 compares the results obtained for the actuated controller with those obtained for the greedy control (GC) algorithm operating the central intersection. The number of vehicles refers to the number of vehicles that departed in each of the eight movements during the simulation period. Summary statistics for the entire intersection are given in Table 2.

Table 1: GC test results.

Movement	No. of vehicles		Tot. delay		Av. delay		Max. queue	
	FAC	GC	FAC	GC	FAC	GC	FAC	GC
1	28	22	1060.1	758.5	37.9	34.5	11	11
2	22	18	752.9	990.9	34.2	55.1	12	14
3	22	27	677.9	1180.6	30.8	43.7	13	11
4	24	37	879.5	942.4	36.6	25.5	15	14
5	18	15	746.6	517.8	41.5	34.5	10	9
6	23	25	693.4	868.9	30.1	34.8	11	12
7	16	13	471.1	626.3	29.4	48.2	12	12
8	20	14	491.3	625.1	24.6	44.7	11	9

Table 2: GC summary statistics.

Statistic	FAC	GC
Total No. of Vehicles	173	171
Total Delay	5772.8	6510.5
Average Delay	33.4	38.1
No. of Vehicles Delayed	164	164
Delay per Delayed Vehicle	35.2	39.7

The results indicate that the greedy control algorithm degraded the performance of more movements than it improved. The overall operational characteristics of the intersection show a slight degradation in performance. This degradation could be due to the loss of the phase transitions where one movement is allowed to continue during the transition of the other.

2.3 Block Control Algorithm

The block control algorithm is loosely based on the simple l-block algorithm described and analyzed in [15]. Instead of a non-decreasing block size, we used a variable block size defined by the vehicles present at the intersection, and instead of an optimal scheduling of the blocks, we used the NEMA dual ring.

Thus, the block control algorithm can be thought of as a full-actuated control in which the maximum signal times are determined by the number of vehicles present when the phase is selected.

Table 3 compares the results obtained for the actuated controller with those obtained for the block control (BC) algorithm operating the central intersection. Again, the number of vehicles refers to the number of vehicles that departed in each of the eight movements during the simulation period. Summary statistics for the entire intersection are given in Table 4.

The results indicate that the block control algorithm produced significant improvements in the performance of almost all movements of the intersection at which it was employed. Furthermore, it did not cause significant degradation of performance of movements that were not improved. The overall operational characteristics of the intersection show significant improvements. Based on these results, the block control algorithm is judged to show promise for real-time control applications.

Table 3: BC test results.

Movement	No. of vehicles		Tot. delay		Av. delay		Max. queue	
	FAC	BC	FAC	BC	FAC	BC	FAC	BC
1	28	31	1060.1	697.3	37.9	22.5	11	11
2	22	23	752.9	364.2	34.2	15.8	12	11
3	22	21	677.9	275.6	30.8	13.1	13	8
4	24	31	879.5	291.4	36.6	9.4	15	9
5	18	16	746.6	271.4	41.5	17.0	10	7
6	23	27	693.4	587.2	30.1	21.7	11	9
7	16	16	471.1	503.6	29.4	31.5	12	10
8	20	22	491.3	658.5	24.6	29.9	11	13

2.4 Greedy Block Control Algorithm

The greedy block control algorithm as its name implies is a combination of the greedy control algorithm of section 2.2 and the block control algorithm of section 2.3. We use the simple $K_{2,2}$ algorithm to select the next phase as in the greedy control algorithm, and the number of vehicles present when the phase is selected to determine the maximum signal times as in the block control

Table 4: BC summary statistics.

Statistic	FAC	BC
Total No. of Vehicles	173	187
Total Delay	5772.8	3649.2
Average Delay	33.4	19.5
No. of Vehicles Delayed	164	177
Delay per Delayed Vehicle	35.2	20.6

algorithm.

Table 5 compares the results obtained for the actuated controller with those obtained for the greedy block control (GBC) algorithm operating the central intersection. As in sections 2.2 and 2.3, the number of vehicles refers to the number of vehicles that departed in each of the eight movements during the simulation period. Summary statistics for the entire intersection are given in Table 6.

The results indicate that the greedy block control algorithm produced significant improvements in the performance of all movements of the intersection at which it was employed. The overall operational characteristics of the intersections show even more significant improvements than the already signif-

Table 5: GBC test results.

Movement	No. of vehicles		Tot. delay		Av. delay		Max. queue	
	FAC	GBC	FAC	GBC	FAC	GBC	FAC	GBC
1	28	25	1060.1	292.3	37.9	11.7	11	9
2	22	18	752.9	217.4	34.2	12.1	12	12
3	22	27	677.9	334.4	30.8	12.4	13	8
4	24	24	879.5	443.8	36.6	18.5	15	8
5	18	21	746.6	233.6	41.5	11.1	10	7
6	23	29	693.4	424.2	30.1	14.6	11	9
7	16	20	471.1	251.4	29.4	12.6	12	11
8	20	16	491.3	238.1	24.6	14.9	11	8

icant improvements of the block control algorithm. Based on these results, the greedy block control algorithm is judged to show even more promise for real-time control applications than the block control algorithm.

Table 6: GBC summary statistics.

Statistic	FAC	GBC
Total No. of Vehicles	173	180
Total Delay	5772.8	2435.2
Average Delay	33.4	13.5
No. of Vehicles Delayed	164	166
Delay per Delayed Vehicle	35.2	14.7

3 Open Problems

The main open questions are how well would the block control and greedy block control algorithms work in reality.

References

[1] AT&T, *UNIX System V, release 4: programmer's reference manual*, Prentice Hall, Englewood Cliffs, NJ, 1990.

[2] J.L. Adler, W.W. Recker and M.G. McNally, Using interactive simulation to model driver behavior under ATIS. in *Proceedings for the 4th Interna-*

tional Conference on Microcomputers in Transportation (J. Chow, ed.). ASCE. New York, 1993, pp. 344–355.

[3] H. Al-Deek, S. Ishak and A.E. Radwan, The potential impact of advanced traveler information systems (ATIS) on accident rates in an urban transportation network. in *Proceedings for the 4th Vehicle Navigation and Information Systems Conference.* IEEE. Piscataway, NJ, 1993, pp. 633–636.

[4] H. Al-Deek and A. Kanafani, Modeling the benefits of advanced traveler information systems in corridors with incidents. *Transportation Res. Part C*, **1** (1993), 303–324.

[5] M. Bell, *Future directions in traffic signal control*, Transportation Research Part A, 26 (1992), pp. 303–313.

[6] R.D. Bretherton and G.T. Bowen, Recent enhancements to SCOOT–SCOOT version 2.4, in *Proceedings for the 3rd International Conference on Road Traffic Control.* IEE, London, 1990, pp. 95–98.

[7] D. Bullock and C. Hendrickson, *Roadway traffic control software*, IEEE Transactions on Control Systems Technology, 2 (1994), pp. 255-264.

[8] S. Chiu and S. Chand, Adaptive traffic signal control using fuzzy logic, in *Proceedings for the 2nd IEEE International Conference on Fuzzy Systems.* IEEE, New York, 1993, pp. 1371–1376.

[9] R.E. Fenton, *IVHS/AHS: Driving into the future*, IEEE Control Systems Magazine, 14:6 (December 1994), pp. 13-20.

[10] N.H. Gartner, OPAC: strategy for demand-responsive decentralized traffic signal control, in *Control, Computers, Communications in Transportation* (J.P. Perrin, ed.). Pergamon. Oxford, UK, 1990, pp. 241–244.

[11] J.J. Henry and J.L. Farges, PRODYN in *Control, Computers, Communications in Transportation* (J.P. Perrin, ed.). Pergamon. Oxford, UK, 1990, pp. 253–255.

[12] S. Irani and V. Leung, *Scheduling with conflicts, and applications to traffic signal control*, in Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, 1996.

[13] S. Irani and V. Leung, *Probabilistic analysis for scheduling with conflicts*, in Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, 1997.

[14] S. Irani and V. Leung, *Scheduling with conflicts*, Technical Report UCI-ICS-97-21, Department of Information and Computer Science, University of California, Irvine, 1997.

[15] S. Irani and V. Leung, *Probabilistic analysis for scheduling with conflicts*, Technical Report UCI-ICS-97-22, Department of Information and Computer Science, University of California, Irvine, 1997.

[16] R.M. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, Plenum Press, New York, 1972.

[17] A.J. Khattak, J.L. Schofer and F.S. Koppelman, Commuters' enroute diversion and return decisions: analysis and implications for advanced traveler information systems. *Transportation Res. Part A*, **27** (1993), 101–111.

[18] E. Koutsoupias and C.H. Papadimitriou, *Beyond competitive analysis*, in 35th Symposium on Foundation of Computer Science, IEEE Computer Society Press, 1994.

[19] P.R. Lowrie, *SCATS, Sydney Co-Ordinated Adaptive Traffic System: a Traffic Responsive Method of Controlling Urban Traffic*, Roads and Traffic

Authority, Darlinghurst, NSW, Australia, 1990.

[20] C. Lund and M. Yannakakis, *On the hardness of approximating minimization problems*, in Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, 1993.

[21] V. Mauro and C. Di Taranto, UTOPIA in *Control, Computers, Communications in Transportation* (J.P. Perrin, ed.). Pergamon. Oxford, UK, 1990, pp. 245–252.

[22] C. Pappis and E. Mamdani, A fuzzy logic controller for a traffic junction, *IEEE Trans. Systems Man Cybernet.*, **7** (1977), 707–717.

[23] W.W. Recker, B.V. Ramanathan, X.-H. Yu and M.G. McNally, Markovian real-time adaptive control of signal systems, *Math. and Comput. Modelling*, **22** (1995), 355-375.

[24] F.S. Roberts, *Discrete Mathematical Models with Applications to Social, Biological, and Environmental Problems*, Prentice Hall, Englewood Cliffs, NJ, 1976.

[25] D.I. Robertson and R.D. Bretherton, Optimizing networks of traffic signals in real time–the SCOOT method, *IEEE Trans. on Vehicular Tech.*, **40** (1991), 11–15.

[26] K.E. Stoffers, *Scheduling of traffic lights - a new approach*, Transportation Research, 2 (1968), pp. 199-234.

[27] F.V. Webster, *Traffic Signal Settings*, Her Majesty's Stationery Office, London, 1958.

[28] R.L. Wilshire, *Traffic signals*, in Traffic Engineering Handbook, Prentice Hall, Englewood Cliffs, NJ, 1992.

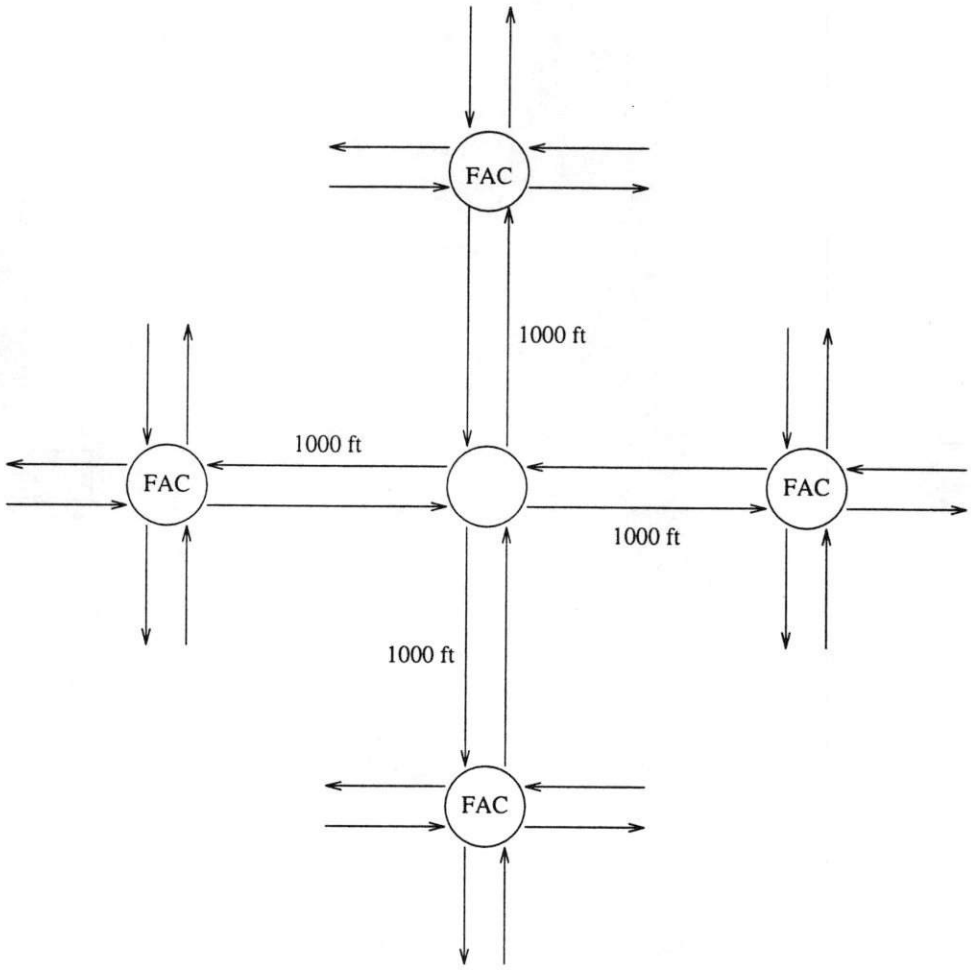

Figure 2: Test network.

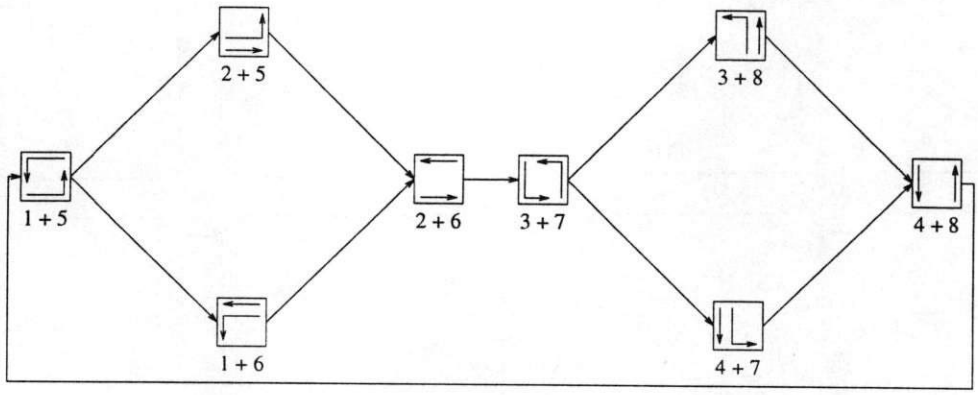

Figure 3: NEMA dual ring controller operation.