

History-based Traffic Control

Gabriel Balan
George Mason University
Fairfax, VA (USA)
gbalan@cs.gmu.edu

Sean Luke
George Mason University
Fairfax, VA (USA)
sean@cs.gmu.edu

ABSTRACT

What if traffic lights gave you a break after you've spent a long time waiting in traffic elsewhere? In this paper we examine a variety of multi-agent traffic light controllers which consider vehicles' past stopped-at-red histories. For example, a controller might distribute credits to cars as they wait and award the green light to lanes with the most credits, allowing cars to keep the credits they accumulate during travel. Such history-based controllers are intended to provide a kind of global fairness, reducing the variance in mean time spent waiting at lights during trips. We compare these controllers against other multi-agent controllers which only consider present information, and discover, among other things, that while the history-based controllers are among the most robust, they often unexpectedly provide more efficiency than fairness.

Categories and Subject Descriptors

I.2.11 [Distributed Artificial Intelligence]: Miscellaneous

General Terms

Algorithms

Keywords

fairness, efficiency-fairness tradeoff, urban traffic control, traffic light controller.

1. INTRODUCTION

You have just spent a long time downtown stopped in gridlocked traffic. As you travel home, shouldn't traffic light controllers recognize this fact and award you the green light, to spread the pain more evenly? This is the idea behind history-based traffic control. Such a traffic controller would take into consideration a vehicle's past stopped-at-red history elsewhere, increasing the fairness with which vehicles are dealt globally on a per-trip basis.

Fairness can be a highly desirable trait: citizens of a town may desire a system which attempts to give every driver approximately the same percentage of waiting-time. But with a few exceptions

[13, 15, 18], the literature has instead largely focused on some notion of system-wide *efficiency*, which might translate into lower emission levels, higher average speed, lower probability of stopping, etc. We imagine this lack of study is because fairness is not an easy thing to implement with existing hard-wired traffic controllers. But if one instead views these controllers in a multi-agent context, with vehicles (and their histories) traveling from controller-agent to controller-agent, then a variety of history-based methods become evident which we will argue notionally provide a kind of fairness. In this paper we compare various multi-agent traffic controllers, including history-based ones, and examine the degree to which they are fair and/or efficient.

The idea behind multi-agent traffic control is to develop a good emergent macro-level traffic-control behavior out of the micro-behaviors of the individual traffic lights. But traffic lights typically "live in the now": they know little of where specific vehicles have been and where they plan to go. At best, such controllers may use past historical information about vehicles in the aggregate, and perhaps heuristic information about likely future traffic patterns. Some past work [21, 22] has explored what happens when traffic light controllers are instead told of trip plans by the drivers: but we think it may not be reasonable to demand such information of the drivers. However, it is plausible to collect information about vehicles' immediate past histories to help inform the controllers, given reasonable privacy restrictions. This allows traffic controllers to communicate with one another, in some sense, through the immediate histories of cars as they travel from intersection to intersection.

We introduce this notion of historical fairness by allowing cars to store "credits" they receive when waiting at red lights, and cash the credits in when passing through intersections. Traffic lights base their decisions on the credits of various cars at the intersection. We also examine history methods which consider the mean waiting time the vehicles have spent at intersections so far along their respective trips. We compare these various methods against two non-history-based methods from the literature: one awards the green light to the lanes with more cars, and the other one prefers the lanes where cars have spent the longest time waiting.

In focusing on the effect of using "history" from a multi-agent perspective, our simplified experimental setup ignores a variety of important game theoretical and modeling issues which may be useful to study in the future. For example, we treat drivers as non-adaptive entities, while real drivers have varying risk attitudes and exhibit adaptive behavior, all of which present complex, difficult-to-model issues [14, 16, 19]. Even so, we believe that the concept of credits as history may be useful in the abstract in various other multi-agent scenarios involving, among other things, network resource allocation [23], package delivery, etc.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

As we discuss later in the paper, our history-based traffic control methods perform well, but often in surprising ways. Notably, though they were designed to provide fairness, several unexpectedly reside in the *efficiency* side of the efficiency-fairness spectrum!

The paper is organized as follows: Section 2 discusses related work. Section 3 introduces our simulation model. Section 4 describes the various traffic-control micro-behaviors in the study. The experiments, comparison metrics, and experimental results are given in Section 5, followed by conclusions in Section 6.

2. RELATED WORK

In this paper we will refer to the traffic controller of an entire intersection as a *traffic light*. A *traffic light pattern* is a combination of red and green lights for each incoming lane that prevents cars from crashing into one another. The interval of time allotted to a pattern is called a *phase*.

The most common metrics used in the traffic light literature are *waiting time* [1, 8, 9, 12, 18, 22], *delay* (the difference between the minimum possible travel time and the actual travel time, including acceleration and deceleration delays) [5, 15, 20], and *travel time* [1, 8, 9, 15]. Other popular metrics are *percentage of stopped cars* [11, 20], *density of cars* [3, 7, 6, 10] and *global flow* [2] (global density times average speed).

One may view efficiency and fairness as the mean and variance, respectively, of these various metrics. Nearly all the literature has concentrated on efficiency (the mean); and fairness has been only lightly studied. Exceptions include: Liu et al [15] analyzed their controllers' efficiency and fairness separately; and Krogh et al [13] and Montana and Czerwinski [18] combined the two into a single measure.

The notion of "fairness" used in Liu et al deals with cars waiting at a single isolated intersection. The cars' delays are averaged per lane¹ and the fairness is the standard deviation of the lanes' expected delays. We instead are interested in fairness (and efficiency) spread throughout a multi-traffic-light system. Thus our efficiency and fairness measures will be largely based on the mean of waiting times at the intersections along a vehicle's trip. This is the "trip waiting time" metric used in [21, 22], but normalized by the number of intersections in the trip.

It is worthwhile mentioning a very popular paradigm in urban traffic control, the "green wave," which coordinates traffic lights so that cars traveling at a certain speed go across those intersections without stopping. The green-wave phenomena cannot be achieved along all streets simultaneously, so traditionally it is used for heavy-flow arteries with light flow intersecting streets. A more general approach is to dynamically decide the sets of intersections that should coordinate. For example in [6] the traffic lights run a distributed constraint satisfaction algorithm to get ad-hoc green-waves.

There have been some efforts to achieve green-waves with implicit communication through density or flow. The idea is to provide an emergent green wave through coordinated behaviors. Bazzan's [3] traffic lights are self-interested agents that learn to coordinate with their neighboring traffic lights by playing coordination games. Gershenson's [11] basic control method changes to green when the total time spent by cars approaching or waiting at the red light exceeds a given threshold. Large enough "platoons" of cars move from light to light, causing it to turn green by their sheer mass, resulting in emergent green waves. This notion is closely related to our new history-based methods presented later.

¹The authors actually used the concept of a *movement*, which is an incoming lane or a group of incoming lanes sending cars in the same direction.

3. THE URBAN TRAFFIC MODEL

We use a network model to simulate traffic flow. Nodes in the network are intersections, each one manned by a traffic light. Edges in the network are one-way road segments, each with a specified length, connecting intersections. Two-way road segments are modeled by using two edges, one in each direction.

On this network travel cars. When a car starts along an edge, it waits for a period of time equal to how long it would take to drive along the road to the next intersection. At that point it emerges and is placed into one of several queues depending on its desired action at that intersection. More formally, for each pair of edges $\langle a, b \rangle$ at an intersection, there is a queue $a \rightarrow b$ into which cars emerging from the edge a are placed while they wait for permission to start traveling on b . Queues may be thought of as the intersections' various turn and go-straight lanes of movement.

Our motivation for using a coarse-grained network of queues, rather than a fine-grained cellular-automaton-style grid of cells, is twofold. First and foremost, it permits the simulation to run much faster. Second, it allows us to choose whether or not to model individual road segments' capacities (we have declined to do so²). We also believe that a similar network structure may be applicable to other relatively unrelated domains, such as entry/exit policies for internet routers [23].

This means that the simulation is neither micro-level nor macro-level. Unlike in [2, 11, 21, 22], we do not explicitly model the movement of cars along road segments. We also make many simplifying assumptions a micro-level model would not make: for example, our cars all run at the same speed regardless of traffic density or road capacity. Further, as we do not consider road capacity, we only examine traffic flow difficulties due to long waits at stoplights. However, we do model cars and traffic lights explicitly, rather than treating them as flow in a macro-level dynamical systems model.

There is a constant number of cars traveling at any time along the streets, each following a predetermined *trip* from an origin intersection to a destination intersection. At the start of the simulation, cars' starting points are randomly chosen from the set of intersections. As a car's trip begins, the car is simply placed on its first edge, and when it completes its last edge, its trip is over (thus the car is not involved in queues at its start or destination). After a car has completed its trip, its former destination becomes the origin of the next trip, and a new destination is chosen at random from among all intersections (excluding the origin). After a trip destination is selected, drivers choose from among two ways to plan the trip route. With 40% probability, drivers will build a trip by incrementally adding to the path a randomly-selected intersection from the set of neighbors that minimize the remaining distance to the destination. With 60% probability, the driver will choose a shortest path with the fewest turns (at most one turn for a two way orthogonal street grid); when there are two such solutions, the driver selects the one whose turn takes place furthest from the city center. The intuition here is that the center of the city is so crowded (mostly with drivers using the first behavior), that there are some drivers who (modeled with the second behavior) prefer to avoid it.³

Intersections. Every twenty seconds (in simulation time), each traffic light individually considers whether to change its pattern. A traffic light may choose between every possible pattern (meaning

²Specifying capacities is more realistic, but unless the drivers are capable of re-routing, capacities can produce artificial deadlocks.

³We recognize that in a more sophisticated scenario, drivers would base routes on previous experience and common places to travel [21, 22].

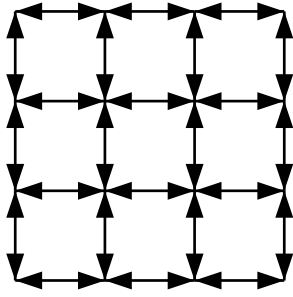


Figure 1: Example (4x4) Street Layout.

maximal set of *non-conflicting* queues). Two queues are said to conflict if cars leaving those queues might collide, either because cars in each queue are entering the same edge, or because the car trajectories intersect with one another. In the intersection $\begin{smallmatrix} \blacksquare & a \\ c & \blacksquare \\ \blacksquare & b \\ \blacksquare & d \end{smallmatrix}$ between road segments a, b, c , and d , the queue $a \rightarrow b$ conflicts with, among others, the queues $c \rightarrow d$ (obviously), $c \rightarrow b$ (because they both enter b), and $b \rightarrow c$ (because, assuming drivers drive on the right-hand side, $b \rightarrow c$ cars turn into the $a \rightarrow b$ traffic). On the other hand, $a \rightarrow b$ does not conflict with $a \rightarrow c$, $a \rightarrow d$, $b \rightarrow a$, $d \rightarrow a$, or $b \rightarrow d$. It also does not conflict with $c \rightarrow c$, $d \rightarrow d$, or $a \rightarrow a$ either, but in this paper paths do not make U-turns, so these queues are always empty.

If a queue awarded the green light was not awarded one in the previous phase (its light was red then), it must wait for 5 seconds while conflicting queues clear the intersection (obtusibly during their yellow light period) before the green light may commence. Thus a light changing from red to green has only a 15-second green period, while a light staying green has a 20-second green period. While a light is green, cars waiting in the corresponding queue are removed one by one and permitted to enter the intersection with a three-second “acceleration penalty” corresponding to getting back up to speed after being stopped at the light. Thus in a 15-second green period five cars can traverse the intersection, but during a 20-second green period six cars can do so. If a car is approaching an intersection and the appropriate queue is empty, the car is allowed to go through at full-speed (with no 3-second penalty). The limit of five (six) cars going through per phase and per lane applies to any combination of cars removed from the queue and cars finding the queue empty and going through full speed.

In our model, all roads are two-way streets, laid out along an orthogonal mesh akin to a downtown city grid (see Figure 1). The blocks are squares, each 0.25 miles on a side. The traffic lights have 0.125 mile sensor ranges, inside which they may discern approaching cars and their desired intersection queues (imagine that turn lanes start 0.125 miles out and sensors are positioned where they start). All cars travel at 25 mph, so they need 36 seconds to cover the segment between two intersections. Regardless of whether they go straight or turn, all cars need exactly 5 seconds to traverse the intersection (not including the acceleration penalty).

4. THE TRAFFIC LIGHT CONTROLLERS

We divide the traffic light controller function into two subfunctions. The first subfunction assigns a *score* to each pattern. The second subfunction selects the winning pattern based on the patterns’ scores.

The second subfunction uses one of two winner-selection methods: a deterministic method and a stochastic method. In the deterministic method, the pattern with the highest total score is the winner; in case of a tie, the winning pattern is the one with most

cars in its queues; further ties are broken at random. The stochastic method uses *roulette selection*: the probability that a pattern will win is proportional to its total score. If all patterns have a score of 0, the pattern with more cars in its queues wins, and further ties are again broken at random. A light only bothers to select a new winner if cars are actually waiting at or approaching the intersection.

The first subfunction is called the *scoring function*, and assigns each pattern a score equal to the total points assigned to cars within the sensor range of the queues receiving the green light in that pattern. We examine the following score functions in combination with the aforementioned winner selection methods:

Counting Cars (CC). Each car yields one point. As a result, this controller defines scores simply as the total number of cars within the queues’ collective sensor range. We include it as a baseline for our comparisons because it is the simplest and most intuitive adaptive traffic controller. Counting Cars was used in [4, 22].

In-Range Time (IRT). A car’s points are equal to the length of time that has transpired since it was picked up by the queue sensors at the current intersection. This is a generalization of Gershenson’s method [11], which only worked with two one-way road intersections. In-Range Time is also closely related to a notion of pheromones applied in [7]. There, cars deposit pheromones as they wait, and the traffic light controller makes a decision based on the total level of pheromones in each lane.

Mean Waiting Time (MWT). This is our first history-based method. The car’s points are equal to its mean time spent waiting at intersections in its trip so far, including the current intersection.

Previous Mean Waiting Time (PMWT). This is our second history-based method. The car’s points are equal to its average waiting time over the previous intersections of the trip but *not* including the current intersection. When at its first intersection of the trip, the car’s score is the global mean waiting time over all intersections in that simulation run. In some sense, $MWT \approx PMWT + IRT$.

Credit Methods (CREDIT). These are our final history-based methods. The car’s points are defined as accumulated “credits” which it receives from, and pays back to, various traffic light controllers. There are four credit-disbursement methods, each a combination of the following two parameters:

- *How are cars compensated for receiving the red light?* If using the *general-sum policy*, after every phase (20 seconds), the cars in the lane of a red traffic light each receive one credit, and cars passing through a green light must pay the traffic light 5 credits or the car’s accumulated credit wealth, whichever is smaller. If using the *zero-sum policy*, after every phase, cars passing through must again pay up to 5 credits as before, and the total credits paid to the intersection at that phase are in turn distributed evenly among cars in red-light lanes at that intersection. If no car is waiting at red, the zero-sum policy will not require a green-light car to pay anything.
- *How many credits does a car have initially?* At the start of a trip, a car’s credit wealth is reset to either 10 or 100 credits.

The values for the parameters of the CREDIT controllers were chosen ad-hoc, and we made no effort to tune them. We note that CREDIT can easily accommodate emergency vehicles and public transportation by endowing special vehicles with unusual numbers of credits, or weighting their credits appropriately.

Grid size	Load	←More Efficient		Pareto Front		More Fair→			
10×10	31	CR(100, G)							
	62	CR(100, G)							
	125	CR(100, G)							
	250	CR(100, G)		IRT					
	500	CR(100, G)		IRT		MWT			
	1K	CR(100, G)		CR(10, G)	IRT	MWT			
	2K	CR(100, G)		CR(10, G)		MWT			
	4K			CR(10, G)	IRT				
	8K	PMWT			CR(10, Z)		IRT	MWT	CR(10, Z)
4×4	16K	PMWT			CR(10, Z)	IRT	IRT		CR(10, Z) CR(10, G)
	31	CR(100, G)		CR(10, G)	IRT				
	62	CR(100, G)		CR(10, G)	IRT				
	125			CR(10, G)	IRT				
	250			CR(10, G)	CR(10, Z)	IRT	MWT		
	500			CR(100, Z)	CR(10, G)	CR(10, Z)	IRT	MWT	
	1K	PMWT					MWT	IRT	MWT
	2K	PMWT			CR(10, Z)		IRT	MWT	CR(10, G)

Table 1: Pareto fronts for all combinations of city size and workload level. The controllers using the stochastic selection are shown with a gray background, while no background is used for the deterministic selection controllers. The CREDIT variants are depicted as CR(E,P), where E is the initial endowment of credits and P is the policy (either Z for zero-sum or G for general-sum compensation).

5. EXPERIMENTS

We tested the traffic light controllers under different city grid sizes and demand levels (what we refer to as the *workload*: the number of cars on the streets). We experimented with two scenarios based on the size of the city grid: 4×4 and 10×10 . As mentioned in Section 3, we used a closed model to generate the workload, so the number of cars was constant throughout a simulation run. In our experiments we chose the following workload levels: 31, 62, 125, 250, 500, 1000, and 2000. For the 10×10 scenario we also used 4000, 8000, and 16000.

All presented results are based on 100 independent runs for each combination of city grid scenario, workload level, and traffic light controller. We ran each simulation for 1000000 seconds (approximately 11.5 days of simulated time). The reason for such a large simulation length is the stochastic nature of the traffic light controllers and route selection, requiring a large sample to accurately compare the traffic light controllers. In order to eliminate the boundary conditions at the beginning of a run, we discarded the data for those trips finished within the first 500 seconds of each run. Simulations were performed using the MASON multi-agent simulation toolkit [17].

5.1 Comparison methodology

When a car reaches its destination, it reports its average waiting time over all intersections along the trip. This statistic is the normalized version of the average trip waiting time (“ATWT”) used in [21, 22].⁴ Of interest to us is the distribution of this statistic over all trips: we use the mean and standard deviation of the distribution as our measures of efficiency and fairness, respectively, in the system.

We used a pareto-dominance criterion to compare two methods: method *A* is superior to method *B* if it is at least as good in both metrics and better in at least one metric. By “better” we mean statistically significantly superior to, and by “at least as good” we will mean significantly superior to or not significantly distinguishable from. Since the efficiency and fairness of a controller are

⁴Using the *total* trip waiting time as our statistic would make drivers on a short trip expect the same total delays as the ones on very long trips, which we felt was unreasonable.

distribution samples, we needed a statistical analysis mechanism to assess the Pareto-dominance relationships. We used unequal-variance repeated t-tests with a Bonferroni-corrected significance level $\alpha = 2.31481 \times 10^{-6}$, which exceeded a 99% confidence in the correctness of the entire set of results presented in this paper.

5.2 Results

Table 1 contains the Pareto fronts for all the city size/workload level combinations. We have taken the liberty of spacing the items in the fronts so that occurrences of the same controller are on the same column. There is no special meaning behind this other than our attempt to make the table easier to read. See Figure 2 for representative examples of what the fronts actually look like.

Overall, the results are very promising for the CREDIT controller, as its variants appear in the Pareto fronts in all our scenarios but one (4×4 grid, 1000 load). In general, CREDIT seems to be a versatile controller, adapting well to both high and low workloads.

An analysis of Table 1 reveals some interesting observations. The most interesting one is that although we expected the history-aware controllers to be fair at the cost of efficiency, some variants proved to be more efficient than the history-oblivious controllers. Most unexpected is PMWT being the most efficient controller for the highest workloads. We did not expect this because PMWT ignores the waiting time information about the current intersection, which would provide efficiency, and uses only past information, which in some sense is the source of fairness. Also, we were surprised to find that the deterministic CREDIT variants are more efficient than MWT, even though the information used by the latter is more accurate than that used by the former.⁵ Oddly, the situation is reversed in the stochastic versions of the controllers.

Another observation is that for (and only for) high enough workload levels, the stochastic selection controllers are consistently

⁵The CREDIT controller cannot tell if a car has waited $k \times 20 + 2$ seconds or $k \times 20 + 18$ seconds. Also, if a queue is long enough, some of the cars waiting in it do not go through in a single 20-second green phase, yet are not compensated for waiting. Therefore a car’s compensation is only a rough measure of the time spent waiting in the queue, as it ignores these “green phases.”

more fair than the deterministic ones (see Figure 2(c) for a typical example). We observe this behavior starting at 1000 cars for the 4×4 grid and 8000 cars for the 10×10 grid. For these scenarios the drivers' expected waiting times per intersection are 2 and 3 minutes respectively, which we feel is not unrealistically high for a rush-hour traffic-jam situation. Why stochastic controllers would perform in this way is still unknown to us. Figures 2 reveal why stochastic controllers disappear from the front however: as workload drops, the deterministic controllers maintain their efficiency advantage but secure more fairness relative to the stochastic controllers.

There are a few other points worth mentioning. One can make a CREDIT variant more fair or more efficient by simply adjusting the initial endowment of credits. The larger the initial number of credits, the less likely a queue with fewer cars can outscore a queue with more cars, resulting in a less fair controller. At the limit, CREDIT with an endowment of infinity is roughly equivalent to CC, except for the tie situations. But CC is dominated in all the scenarios presented here, which leads us to the conjecture that fair tie breaking should not be ignored when designing a traffic light controller.

6. CONCLUSIONS

We have introduced the concept of trip-based fairness and showed a simple approach to incorporate it into traffic light control. We have implemented several controllers to investigate the effect of awarding the green light based on information regarding the cars' past experiences. The testbed we used was a meso-simulated two-way street grid scenario with uniform demand and a wide range of traffic loads. All controllers investigated here, however, work out-of-the-box with any intersection set-up: arbitrary number and layout of one- and two-way streets, with and without shared lanes. Cars are not required to participate in the traffic control system: if a light senses a non-participating car, it might set up a proxy credit accumulator initiated with some mean credit amount, but which does not travel with the car when it leaves the intersection.

We expected the newly proposed controllers to be fair at the expense of efficiency, but they performed reasonably well in general, and some even turned out unexpectedly efficient (at the expense of fairness). More research is needed to gain a fuller understanding of the phenomena involved.

The data provided interesting (and often unexpected) results when dealing with overcrowded traffic conditions. Here, stochastic selection is a viable solution for improving fairness; and for the history-based methods, the cruder the use of information about the present, the more efficient the controller.

6.1 Future work

Presently our controllers work with a pre-set phase length. While this is not an unusual bias in the traffic light control literature, we think we should try to eliminate the phase length parameter or incorporate phase adaptation into the controllers.

The original inspiration for this paper was to allow the drivers to let the traffic lights know whether the drivers are in a hurry or not. We plan to investigate various auction-style methods for traffic light controllers. In these methods, the drivers are self-interested agents trying to resolve who goes first using a currency accepted at all traffic lights. The lights act as benevolent auctioneers, awarding green lights to those deserving it (able and willing to "pay") the most. When drivers are not in a rush they might choose to bid low and save some credits for the future. The resulting traffic controller system acts as a market, which hopefully requires less tuning because it is self adaptive.

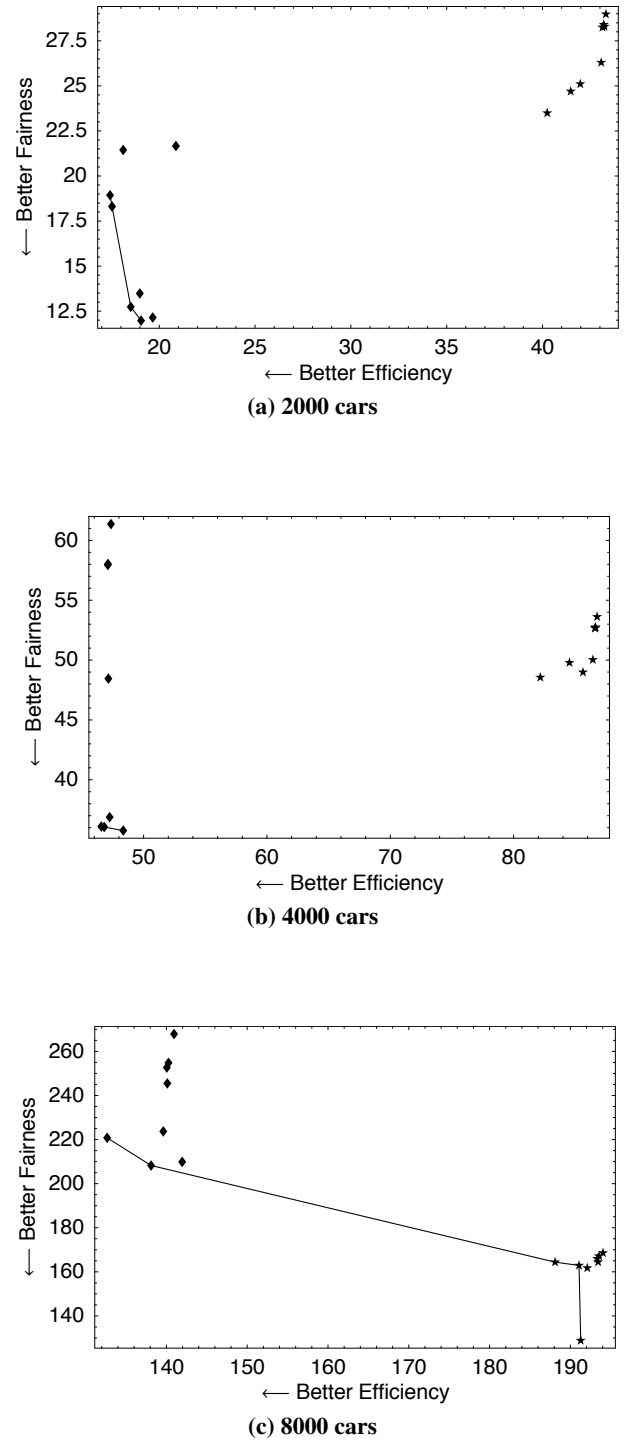


Figure 2: Results for the 10×10 grid with different workloads: 2000, 4000 and 8000 cars. The deterministic controllers are represented as $\diamond$ and the stochastic ones as $\star$; the lines connect the non-dominated controllers (see Table 1); the units on the plots' axes are seconds.

This is only a sample of future work for this domain. We think distributed traffic control is an attractive problem for multi-agent systems, with a broad, relatively unexplored research domain and a payoff in the form of immediate real-world impact. The area also presents a number of major research challenges: the number of cars and traffic-control devices on the road is huge, and the problem is highly decentralized with regard to vehicles and, more often than not, with regard to traffic-control devices. The humans in the vehicles co-adapt to one another and to traffic controllers. Dense traffic can produce emergent, unexpected phenomena. As such, we imagine this area may prove both fruitful and increasingly important in the future.

7. ACKNOWLEDGMENTS

We are grateful to Aimee Flannery for her invaluable assistance, and to Liviu Panait for his help in organizing this paper.

8. REFERENCES

- [1] J. M. Anderson, T. M. Sayers, and M. G. H. Bell. Optimization of a fuzzy logic traffic signal controller by a multiobjective genetic algorithm. In *Ninth International Conference on Road Transport Information and Control*, pages 186–190, April 1998.
- [2] R. Barlović, T. Huisinga, A. Schadschneider, and M. Schreckenberg. Adaptive traffic light control in the ChSch model. In P. H. L. Bovy, S. P. Hoogendoorn, M. Schreckenberg, and D. E. Wolf, editors, *Workshop on Traffic and Granular Flow*, 2003.
- [3] A. L. C. Bazzan. A distributed approach for coordination of traffic signal agents. *Autonomous Agents and Multi-Agent Systems*, 10(2):131–164, March 2005.
- [4] C. Bertelle, A. Dutot, S. Lerebourg, and D. Olivier. Road traffic management based on ant system and regulation model. In *The International Workshop on Modeling & Applied Simulation*, pages 35–43, Bergeggi (Italy), October 2003.
- [5] E. Bingham. Reinforcement learning in neurofuzzy traffic signal control. *European Journal of Operational Research*, 131(2):232–241, July 2001.
- [6] D. de Oliveira, A. L. C. Bazzan, and V. Lesser. Using cooperative mediation to coordinate traffic lights: a case study. In *The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 463–470, July 2005.
- [7] D. de Oliveira, P. R. Ferreira, Jr., P., A. L. C. Bazzan, and F. Klügl. Reducing traffic jams with a swarm-based approach for selection of signal plans. In *Proceedings of Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence*, Brussels, Belgium, 2004. Poster.
- [8] K. Dresner and P. Stone. Multiagent traffic management: An improved intersection control mechanism. In *The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 471–477, July 2005.
- [9] N. V. J. Findler and J. Stapp. A distributed approach to optimized control of street traffic signals. *Journal of Transportation Engineering*, 118, 1992.
- [10] J. France and A. A. Ghorbani. A multiagent system for optimizing urban traffic. In *IAT*, pages 411–414. IEEE Computer Society, 2003.
- [11] C. Gershenson. Self-organizing traffic lights. *Complex Systems*, 2005. to appear.
- [12] I. Kosonen. Multi-agent fuzzy signal control based on real-time simulation. *Transportation Research Part C*, 11(5):389–403, October 2003.
- [13] C. Krogh, M. Irgens, and H. Traetteberg. A novel architecture for traffic control. In L. Olaussen and E. Helli, editors, *The 3rd International Conference on Vehicle Navigation and Information Systems*, pages 75–81, 1992.
- [14] D. Levinson. The value of advanced traveler information systems for route choice. *Transportation Research Part C*, 11(1):75–87, February 2003.
- [15] H. X. Liu, J.-S. Oh, and W. Recker. Adaptive signal control system with on-line performance measure. *Transportation Research Record*, 1811:131–138, 2002.
- [16] H. X. Liu, W. Recker, and A. Chen. Uncovering the contribution of travel time reliability to dynamic route choice using real-time loop data. *Transportation Research Part A*, 38(6):435–453, July 2004.
- [17] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan. Mason: A multi-agent simulation environment. *Simulation*, 81(7):517–527, July 2005.
- [18] D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 333–338, Stanford University, CA, USA, 1996. MIT Press.
- [19] J. Wahle, A. L. C. Bazzan, F. Klgl, and M. Schreckenberg. The impact of real-time information in a two-route scenario using agent-based simulation. *Transportation Research Part C*, 10(5–6):399–417, December 2002.
- [20] W. Wei and M. Wang. Traffic signal control using fuzzy and neural network. In *8th International Conference on Neural Information Processing*, Shanghai, China, November 2001.
- [21] M. Wiering, J. van Veenen, J. Vreeken, and A. Koopman. Intelligent traffic light control. Technical Report UU-CS-TR-2004-029, Institute of Information and Computing Sciences, Utrecht University, 2004.
- [22] M. Wiering, J. Vreeken, J. van Veenen, and A. Koopman. Simulation and optimization of traffic in a city. In *IEEE Intelligent Vehicles Symposium (IV'04)*, 2004.
- [23] Y. Zhou. *Resource Allocation in Computer Networks: Fundamental Principles and Practical Strategies*. PhD thesis, Drexel University, 2003.