Distributed Learning of Lane-Selection Strategies for Traffic Management

David E. Moriarty
Pat Langley
Intelligent Systems Laboratory
Daimler-Benz Research and Technology Center
1510 Page Mill Road, Palo Alto, CA 94304
moriarty, langley@rtna.daimlerbenz.com

Abstract

This paper explores a novel approach to traffic management that relies on a distributed scheme in which vehicles themselves select lanes. After formulating the problem of distributed traffic control, we describe an initial system that uses reinforcement learning to acquire lane selection strategies from experience with a traffic simulator. In addition, we report experimental studies of this method which demonstrate that the learned strategies let drivers more closely match their desired speeds than do handcrafted strategies and that it also reduces the number of lane changes. The experiments also suggest that the learned behaviors generalize to different traffic densities, different numbers of lanes, situations involving blocked lanes, and even the presence of selfish drivers. In addition, we report lesion studies that reveal the contributions of the different learning modules. In closing, we discuss related work on traffic control and consider some directions for future research.

Contact author: Pat Langley; langley@rtna.daimlerbenz.com; (650) 845-2532

Running head: Distributed Learning for Traffic Management

Key words: reinforcement learning, distributed control, traffic management, lane selection

1 Introduction

A large effort is under way by government and industry in America, Europe, and Japan to develop intelligent vehicle and highway systems (IVHS). These systems incorporate ideas from artificial intelligence, intelligent control, and decision theory, among others, to automate many aspects of driving and traffic control. The goals of IVHS are quite broad and include increased traffic throughput, fewer accidents, reduced fuel consumption, and a better driving experience.

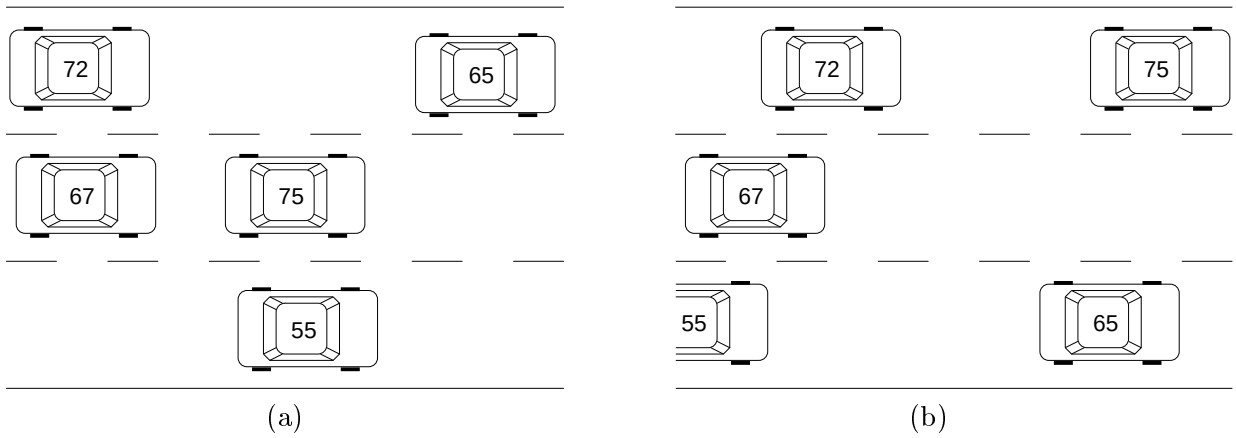
The work in this paper targets one component of the overall task: the problem of managing traffic. Advanced traffic management systems (ATMS) are designed to reduce congestion and increase overall traffic throughput. Most such systems maintain efficient traffic flows by controlling traffic signals and highway ramp meters, treating traffic as a single mass and ignoring the behavior of individual cars (Gilmore, Elibiary, & Forbes, 1994; Kagolanu, Fink, Smartt, Powell, & Larson, 1995; Pooran, Tarnoff, & Kalaputapu, 1996). Another technique, actually used on London's beltway, calculates a fixed speed for each lane that would give efficient traffic flow, then assigns each vehicle to a lane and requires them to proceed at the indicated speed (Hall, 1995; Ramaswamy, Medanic, Perkins, & Benekohal, 1997).

However, these approaches miss an important component of traffic management: coordination of the cars themselves. Drivers generate local behaviors such as lane changes and speed control, and these behaviors could be coordinated and optimized to better maintain desired speeds and achieve greater traffic throughput. This suggests that a challenging problem for machine learning lies in the development of cooperative driving strategies for traffic management. This paper explores one form of this problem: intelligent lane selection. Each car receives local input of the surrounding traffic patterns and the desired speed of the driver and outputs the lane in which to drive. A car's lane selections should consider not only the maintenance of its own desired speed, but also how the selection will affect the speeds of other cars. In this way, the cars should organize themselves into a cooperative system that lets the fast drivers pass through, while still letting the slow drivers maintain their speeds.

This work in this paper is exploratory in nature, and follows Dietterich's (1990) model for exploratory machine learning research. We formulate the novel problem of traffic management from a car-centered, machine learning perspective and present initial results of cooperative lane selection. The next section recasts traffic management as a search for cooperative controllers in a distributed environment and motivates the application of machine learning. Section 3 describes our learning system and section 4 presents empirical results in a simulated traffic environment. Section 5 discusses related work, followed by an outline of some important future research directions in section 6. The final section summarizes our work and gives some concluding remarks.

2 Car-Centered Traffic Management

As noted earlier, our approach to traffic management involves a reformulation of the problem into a distributed artificial intelligence task, in which cars coordinate lane changes to maintain desired speeds and reduce total lane maneuvers. We define the problem with no assumptions about the level of automation of the cars. Lane selection information could be provided to the driver, who completes the maneuver, or to a regulation controller in an automated car (Pomerleau, 1995; Varaiya & Shladover, 1991; Varaiya, 1993). However, our view is that intelligent lane selection will be most beneficial for manual driving, since automatic driving normally assumes that cars will



(a) (b)

Figure 1: (a) An example traffic situation in which the traffic flows from left to right and the number on each car shows the car's speed. (b) Traffic after reorganization in which car 75 and 65 swap lanes followed by another lane change by car 65, so that all cars can maintain their desired speeds.

travel at common speeds in platoons and thus there is less need for reorganization of the cars based on differing individual speeds.

2.1 Problem Definition

In the current advanced traffic management view, cars are considered tokens that follow simple, selfish rules of behavior. These management systems affect the flow of the car tokens by controlling external, fixed-position devices such as traffic signals, ramp meters, speed limits, and dynamic lanes. Surprisingly, very little research has addressed how the cars themselves can sense and intelligently affect traffic dynamics. One exception is the work of Carrara and Morello in the DOMINC project.

Our view is that cars are not blind tokens, but rather can sense their environment and act intelligently and cooperatively to achieve a desired global behavior. More specifically, cars can learn to organize themselves by traffic lanes to increases overall traffic throughput, reduce the average number of lane changes, and maintain the desired speeds of the drivers. Intelligent lane selection should therefore complement existing efforts in advanced traffic management by providing better throughput in between traffic signals and better-defined driver behaviors for traffic-flow prediction.

Figure 1(a) illustrates a situation where lane coordination is beneficial. The figure illustrates five cars along with their speeds, which will be used as identifiers. Car 72 is quickly approaching car 65 and will be unable to pass because of the position of car 67. Without reorganization, car 65 forces car 72 to reduce its speed and wait for car 67 to pass car 65, which will decrease traffic throughput and car 72's satisfaction. An efficient solution to this problem is for car 75 and car 65 to immediately swap lanes, followed by car 65 moving into the bottom lane, as shown in Figure 1(b). This maneuver ensures that no speeds are reduced and no throughput is lost.

We recast the traffic management problem as a problem in distributed artificial intelligence, where each car represents an individual agent in a multi-agent system. Cars act on their world (highway) by selecting appropriate lanes to drive in. They interact with other cars by competing for resources (spaces or *slots* on the highway). Each action is local in nature, and may not produce any noticeable benefit to the car. Collectively, however, the local actions can improve the global

performance of the traffic. For example, yielding a lane to a faster car does not produce any local benefit to the slower car, but does increase the overall traffic throughput and let the passing car maintain its desired speed.

Global traffic performance could be defined in many different ways. Governments want high traffic throughput, whereas drivers want to maintain desired speeds with few lane changes. We selected the driver-oriented metric, since drivers are likely to be the harshest critics of cooperative driving. The performance function P we devised for a set of cars C is given by the equation:

$$P(C) = \frac{\sum_{t=1}^{T} \sum_{i=1}^{N} (S_{it}^{a} - S_{it}^{d})^2}{TN} - \frac{60 \sum_{i=1}^{N} L_i}{TN}, \quad (1)$$

where T is the total time steps (in seconds), N is the number of cars, S_{it}^{d} is the desired speed of car i at time t, S_{it}^{a} is the actual speed of car i at time t, and L_i is the total number of lane changes for car i over T time steps. The goal is to minimize the difference between actual speeds and desired speeds averaged over several time steps and over all cars on the road. Each speed difference is squared to penalize extreme behavior. For example, driving 60 m/h 90% of the time and 10 m/h 10% of the time gives an average of 55 m/h but is clearly less desirable than driving 56 m/h 50% and 54 m/h 50% of the time, which also gives an average of 55 m/h. Squaring the error from desired speed gives a higher evaluation to the more consistent strategy. To discourage excessive lane changes, the performance function is adjusted by subtracting the number of lane changes per minute averaged over all cars.

The problem is thus to find a lane-changing strategy or a set of strategies to maximize equation 1. A naive strategy for each car, which most traffic management systems assume, is to select the lane that lets it most consistently achieve its desired speed and only change lanes if a slower car is encountered. The disadvantage of such a strategy is that it does not take into account the global criteria of traffic performance. A slow car should not drive in the "fast" lane simply because it can maintain its desired speed. We will refer to cars that employ the naive strategy as *selfish cars*, since they maximize the local performance of their respective car. We are interested in strategies that maximize the aggregate performance of traffic. Cars that employ cooperative lane-selection strategies will be termed *smart cars*.

Ideally, the smart cars should coexist with current drivers on the highways. This situation poses interesting research questions. How many smart drivers are necessary to make cooperation worthwhile? How quickly does the system break down when selfish drivers are introduced in the system? Section 4 presents some empirical evidence that, even in traffic distributions as high as 95% selfish cars, cooperative lane selection can improve traffic performance.

2.2 Controller Communication and Coordination

The previous section defined the problem of car-centered traffic management, but left open some important issues in designing a distributed artificial intelligence system for traffic management. Specifically, we left open the level of communication between the cars and the amount of knowledge available on other cars' decisions and state. The multi-agent literature is often divided on these matters, and we feel that it is not central to the problem definition. Here we describe our assumptions about communication and state information.

We assume that cars have access to information on their own state, including knowledge of their current driving speed and the driver's desired speed. One could imagine a driver specifying desired

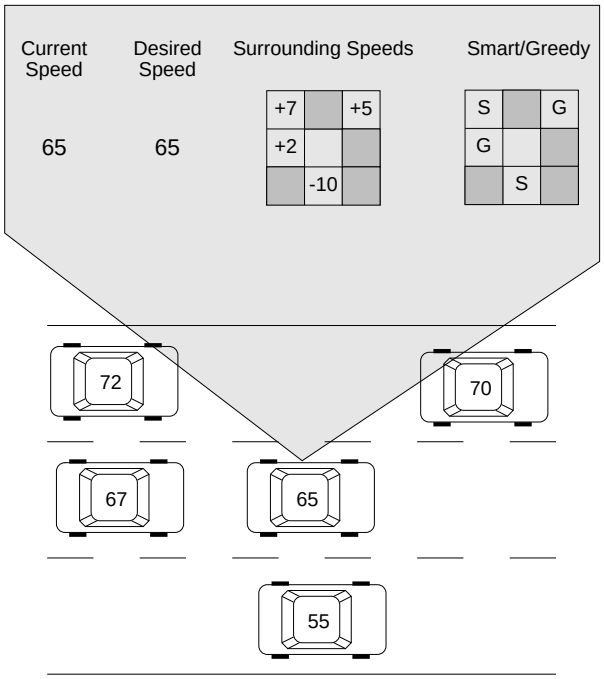


Figure 2: An illustration of the input to each car. The shaded region shows the current input information for the middle car. The car has access to its current speed, its desired speed, the relative speeds of surrounding traffic, and whether other cars are smart or selfish.

speeds at the start of a trip, or the system could infer this information from the driver's historical behavior. We also assume that cars can perceive limited state information of surrounding cars, such as their relative speeds. The system could sense this information using radar or receive it directly from other cars via radio waves or the Internet. Cars should also sense which surrounding cars are cooperative and which are selfish. Again, the system could infer cooperation from driver behavior or direct communication.

Figure 2 illustrates the input for a car in a specific traffic situation. The middle car receives as input its current speed, its desired speed, the relative speeds of surrounding traffic, and whether surrounding cars are cooperative or selfish. The range and granularity of the relative speed inputs could be adjusted to take into account both local traffic and upcoming traffic. For example, it may prove beneficial to receive not only relative speeds of individual cars in the immediate vicinity, but also relative speeds of groups of cars in more distant ranges.

We assume that the controller's output consists of three options: (1) stay in the current lane, (2) change lanes to the left, or (3) change lanes to the right. The output does not specify the best lane to drive in, but rather whether the lanes immediately left or immediately right are better than the current lane. This control provides flexibility, since it does not depend on the number of lanes on the roadway or knowledge of the current driving lane. Thus, controllers that learn on a three-lane highway should, at least in principle, generalize to greater or fewer lanes.

We assume that the controller's output represents a ranking of the three possible choices, with the highest ranked choice that is both valid and safe being selected as the car's next action. For a recommendation to be valid, there must be a lane available in the specified direction. For a recommendation to be safe, there must not be a car in the same longitudinal position in the new lane. It is always safe to remain in the current lane. The system could also incorporate other safety

assurances such as detecting whether a lane change produces an unsafe spacing between cars in the new lane. For example, one might specify that a very slow car should not move in front of a very fast car even if there is immediate space for it in the fast car's lane, since the fast car will likely close that space during the span of the lane change.

The higher-level safety/validation process relieves the controller of the overhead of deciding what lanes are available and safe and centers the control problem on the specific area of concern: which lanes are better. In other words, by removing the problem of validation and safety, the controller can focus on and more easily learn the task of lane ranking. This approach is analogous to removing legal move identification in game playing.

Another important issue concerns the representation of the different lane-selection strategies. Clearly, different types of drivers should select lanes differently. Slower drivers will normally (but not always) use lane selection to open up lanes for faster traffic, whereas faster drivers will select lanes to get through slower traffic. Average-speed drivers will employ elements of both strategies. At issue is how the different types of strategies are represented and implemented.

One approach is to maintain an explicit control policy for each type of driver. For example, fast drivers would utilize the fast lane-selection strategy and slow drivers the slow lane-selection strategy. A disadvantage of this approach is that it requires *a priori* knowledge of the number of driver types and the boundaries that separate them. Also, it does not provide a smooth transition between styles of driving. A driver on a boundary would be forced into one of the two surrounding strategies instead of an interpolation between the two.

A better approach is to parameterize the driving style and use it as input to a single control policy. Each car would contain the same control policy, but since it receives driving style as input, it behaves differently for different types of drivers. In this case, driving style is simply the desired speed. No a priori decisions are necessary regarding the number of lane-selection strategies or their boundaries. Moreover, since the different strategies are keyed to a continuous input (desired speed), there can be smooth transition and interpolation between different lane-selection strategies.

2.3 Learning Distributed Control Strategies

Creating distributed lane-changing controllers by hand appears quite difficult. It is unclear whether experts exist in this domain and, even if they do, experts often find it difficult to verbalize complex control skills, which creates a *knowledge acquisition bottleneck*. Also, the innumerable traffic patterns and varying driving styles create a very large problem space. Even with significant expert domain knowledge, hand crafting a controller that operates effectively in all areas of the problem space may not be feasible.

Another solution is to apply machine learning to develop intelligent controllers through direct experience with the domain. A learning algorithm would modify the controller based on good and bad experiences in the problem space. This approach frees us from the task of acquiring and encoding expert domain knowledge, since it discovers examples of good and bad decisions through direct experience. Moreover, the controllers are not necessarily fixed and could continue to learn and adapt with new experiences.

The lane-selection problem appears out of reach of the more standard, *supervised* machine learning methods (e.g., Quinlan, 1986; Rumelhart, Hinton, & Williams, 1986). In supervised learning, control policies are formed from examples of correct behavior. In the case of intelligent lane selection, supervised learning requires demonstrations of good and bad lane selections. Without

expert domain knowledge it is difficult to generate these examples. In many control problems, supervised learning is used to mimic the behavior of people (e.g., Pomerleau, 1992; Sammut, Hurst, Kedzier, & Michie, 1992). For intelligent lane selection, however, this is exactly what we do not want to model. We believe that most drivers do not select lanes intelligently, but are rather more selfish in nature. Thus, it seems erroneous to use real driver behaviors as a basis for learning cooperative lane selection.

A more flexible machine learning approach that is capable of learning from general rewards instead of behavioral examples has been termed *reinforcement learning*. The rewards provide only a general measure of proficiency over the task and do not explicitly direct the learner towards any course of action. The learner adjusts its actions through trial and error interactions with the underlying system to maximize the reward signal. In the lane-selection problem, the cars can receive rewards by solving equation 1 at specific time stepqs. Cars then adjust their lane-selection strategies using some reinforcement learning algorithm to maximize the reward function. There are two main approaches to the task of reinforcement learning: methods that learn through temporal differences (Sutton, 1988; Watkins & Dayan, 1992; Kaelbling, Littman, & Moore, 1996), and methods that learn through evolutionary algorithms (Grefenstette, Ramsey, & Schultz, 1990; Holland & Reitman, 1978; Moriarty & Miikkulainen, 1996a; Whitley, Dominic, Das, & Anderson, 1993; Wilson, 1994). This paper adopts an evolutionary algorithm as the primary reinforcement learning mechanism, but also employs a technique similar to temporal difference learning for smaller strategy refinements.

3 An Approach to Learning Lane-Selection Strategies

To test our hypothesis that machine learning can produce effective lane-selection strategies for traffic management, we developed a learning approach tailored to this domain. The learning system consists of three main components: reinforcement learning using the SANE neuro-evolution system, supervised learning from pre-existing domain knowledge, and a local learning strategy that is similar in spirit to temporal difference methods. Figure 3 illustrates the interaction of the different learning methods, which are described in the next three sections.

3.1 Reinforcement Learning using SANE

The backbone of the learning system is the SANE reinforcement learning method (Moriarty & Miikkulainen, 1996a; Moriarty, 1997). SANE (Symbiotic, Adaptive Neuro-Evolution) was designed as a fast, efficient method for forming decision strategies in domains where it is not possible to generate training data for normal supervised learning. The system maintains a population of possible strategies, evaluates the goodness of each from its performance in the domain, and uses an evolutionary algorithm to generate new strategies. The evolutionary algorithm modifies the pool of strategies through common genetic operators like selection, crossover, and mutation (Goldberg, 1989).

SANE represents its decision strategies as artificial neural networks that form a direct mapping from sensors to decisions and provide effective generalization over the state space. The evolutionary algorithm searches the space of hidden neuron definitions, where each hidden neuron defines a set of weighted connections between a fixed input and fixed output layer. In other words, SANE evolves all of the connections and weights between the hidden layer and the input and output layers in a three-layer network.

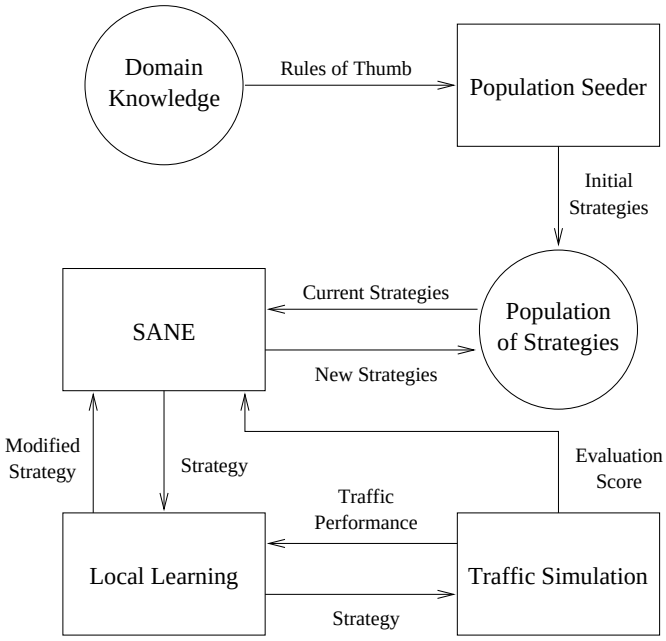


Figure 3: The organization and interaction of the different learning modules.

SANE offers two important advantages for reinforcement learning that are normally not present in other implementations of neuro-evolution. First, it decomposes the search for complete solutions into a search for partial solutions. Instead of searching for complete neural networks all at once, solutions to smaller problems (good neurons) are evolved, which can be combined to form an effective full solution (a neural network). In other words, SANE effectively performs a problem reduction search on the space of neural networks.

Second, the system maintains diverse populations. Unlike the canonical function optimization evolutionary algorithm that converges the population on a single solution, SANE forms solutions in an *unconverged* population. Because several different types of neurons are necessary to build an effective neural network, there is inherent evolutionary pressure to develop neurons that perform different functions and thus maintain several different types of individuals within the population. Diversity lets recombination operators such as crossover continue to generate new neural structures even in prolonged evolution. This feature helps ensure that the solution space will be explored efficiently throughout the learning process. Thus, SANE is more resilient to suboptimal convergence and more adaptive to domain changes than standard evolutionary algorithms (Moriarty, 1997).

SANE represents each lane-selection strategy as a neural network that maps a car's sensory input into a specific lane-selection decision. Figure 4 shows the input and output of the lane-selection networks. Each network consists of 16 input units, 12 hidden units, and 3 output units. A network receives input on the car's current and desired speeds and the speeds of surrounding traffic, and it outputs a ranking of the three possible choices.

A strategy is evaluated by placing it in a traffic simulator and allowing it to make lane selection decisions in a certain percentage of the cars. Each strategy is evaluated independently of other strategies in the population, so that only one is evaluated at a time. Cars not under control of the strategy being evaluated follow "default" rules of behavior. We measure the fitness of a strategy using equation 1 after some number of simulated seconds. SANE uses these evaluations to bias its genetic selection and recombination operations toward the more profitable lane-selection strategies.

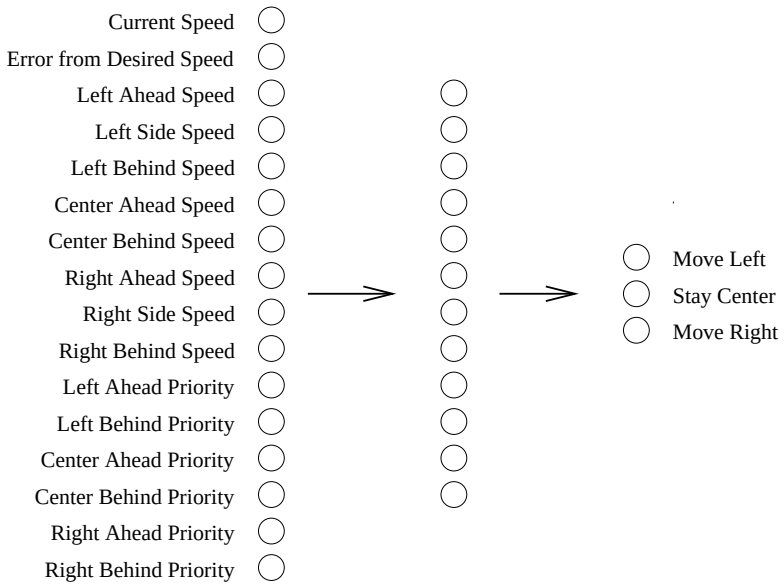


Figure 4: The input and outputs to the neural networks for lane selection.

3.2 Incorporating Existing Domain Knowledge

The second learning component capitalizes on pre-existing domain knowledge and gives SANE a good starting set of initial strategies. Although expert information is difficult to obtain in this problem, general rules of thumb are not. For example, one good rule of thumb specifies that a very slow driver should in general not drive in the far left lane. Supervised learning from these general rules of behavior will not generate optimal lane selection strategies, but it can give the learning system a good head start towards intelligent behavior.

The population seeder applies rules of thumb in the traffic simulator and generates a series of input and output pairs, which represent decisions made from the rules of thumb based on specific sensory input. These pairs denote examples of good behavior that can be fed to a supervised learning method to form initial strategies. Since SANE's strategies are represented as neural networks, the population seeder employs the backpropagation algorithm (Rumelhart et al., 1986) to train the networks over the training examples. To maintain diversity within the initial population of neural networks and not overly bias SANE toward the rules of thumb, only a subset of the networks are seeded using the default knowledge. In practice, we seed 25% of the initial population.

We used four separate rules to seed SANE's initial population of strategies:

- If your desired speed is 55 m/h or less and the right lane is open, then change lanes right
- If you are in the left lane, a car behind you has a higher speed, and the right lane is open, then change lanes right
- If a car in front of you has a slower current speed than your desired speed and the left lane is open, then change lanes left.
- In the previous situation, if the left lane was not open but the right lane is open, then change lanes right.

These rules are based on our interpretation of the "slower traffic yield to the right" signs posted on the highways. We will refer to this strategy hereafter as the *polite strategy*. The selfish strategy described in section 2.1 operates using only the last two rules.

3.3 Local Learning through Performance Differences

We also implemented a local learning module that, like the population seeder, was implemented to increase learning efficiency and thereby reduce the amount of simulation time necessary to form good strategies. Local learning occurs during the evaluation of a lane-selection strategy and makes small refinements to the strategy based on immediate rewards or penalties. A reward or positive training signal is given if there is a significant increase in traffic performance and a penalty or negative training signal is given if there is a significant decrease. Performance is measured at regular intervals. In practice, we sample traffic performance every 10 simulated seconds and generate a reward or penalty signal if the difference in performance from equation 1 is larger than ten.

If a training signal is generated, all actions performed in the sampling interval are considered responsible. If the signal is positive, each of those actions is reinforced. If it is negative, they are punished. Reinforcement and punishment are achieved by backpropagating error signals associated with the network's activation in that situation and a training example derived from the training signal. For example, reinforcement on a *change left* decision would create a training example of the previous input paired with the target output (0.0, 1.0, 0.0). The targets of *stay center* and *change right* are 0.0 and change left is 1.0. Using the standard backpropagation procedure, the weights are updated based on this training example and the resulting network is more likely to choose *change left* in a similar situation. A negative training signal in the previous example, would generate a target output of (1.0, 0.0, 1.0), and the resulting network would be less likely to choose *change left* in similar situations.

The learning strategy is somewhat similar to temporal-difference methods (Sutton, 1988) for reinforcement learning, in that updates are based on the performance differences over successive time periods. However, temporal difference methods treat performance differences as prediction errors from which they can learn to predict future rewards. Our local learning component uses the differences to determine whether to reinforce or penalize specific decisions, but it does not directly address the issue of credit assignment. Our framework could also use a temporal-difference method for local learning, and we expect to evaluate this approach in the near future.

4 Experimental Evaluation

Intelligent lane selection offers important advantages for traffic control and our learning method offers a plausible approach to generating selection strategies, but its actual behavior remains an empirical question. In this section, we report experimental studies of our approach in a simulated traffic environment. We demonstrate that learned strategies for lane selection let drivers more closely match their desired speeds than handcrafted strategies, while also reducing the number of lane changes. We also show that the learned behaviors generalize to different traffic densities, different numbers of lanes, situations involving blocked lanes, and even the presence of selfish drivers. We also report lesion studies that reveal the contributions of the different learning modules.

Earlier in the paper, we characterized this research as exploratory in nature. Following Dietterich's (1990), advice, we have focused our efforts on precisely stating a challenging new problem for machine learning, showing the feasibility of solving this problem by describing an initial system, and illustrating some important issues that arise in experimental studies of such systems. We do not claim that our particular system is superior to other methods of learning strategies for distributed traffic control, and we leave experiments designed to answer such questions for future research.

4.1 A Simulated Traffic Environment

To evaluate traffic management through intelligent lane selection, we developed a simulator to model traffic on a highway. For each car, the simulator updates the continuous values of position, velocity, and acceleration at one second intervals. The acceleration and deceleration functions were set by visualizing traffic performance under different conditions and represent our best estimate of the behavior of actual drivers. Acceleration (A) is adjusted based on the equation $A(s) = 10s^{-0.5}$, where s represents the current speed in miles per hour (m/h).

Deceleration occurs at the rate of -2.0 m/h per second if the difference in speed from the immediate preceding car is greater than twice the number of seconds separating the two cars. In other words, if a car approaches a slower car, the deceleration point is in proportion to the difference in speed and the distance between the cars. If there is a large difference in speed, cars will decelerate sooner than if the speed differences are small. If the gap closes to two seconds, the speed is matched instantaneously. Lane changes are only allowed if the change maintains a two-second gap between preceding and following cars.

The simulated roadway is 13.3 miles long, but the top of each lane "wraps around" to the bottom, creating an infinite stretch of roadway. The simulator was designed as a tool to efficiently evaluate different lane-selection strategies and thus makes several assumptions about traffic dynamics. The primary assumptions in the current model are:

- All cars have the same size and mass;
- All cars use the same acceleration rules;
- Cars accelerate to and maintain their desired speed if there are no slower, preceding cars, and they never exceed their desired speed;
- A lane change takes exactly one time step (one second); and
- There are no curves, hills, on ramps, or exit ramps.

Although none of these assumptions hold for real-world traffic, they are also not crucial to evaluate the merits of intelligent lane selection. Removing these assumptions unnecessarily complicates the model, which creates unacceptable run times for exploratory research. In future work, we will expand our experiments to a more realistic simulator such as SmartPATH (Eskafi, 1996).

During training, the learning system uses the traffic simulator to evaluate candidate lane-selection strategies. Each evaluation or *trial* lasts 400 simulated seconds and begins with a random dispersement of 200 cars over three lanes on the 13.3 mile roadway. Desired speeds are selected randomly from a normal distribution with mean 60 m/h and standard deviation 8 m/h. In each

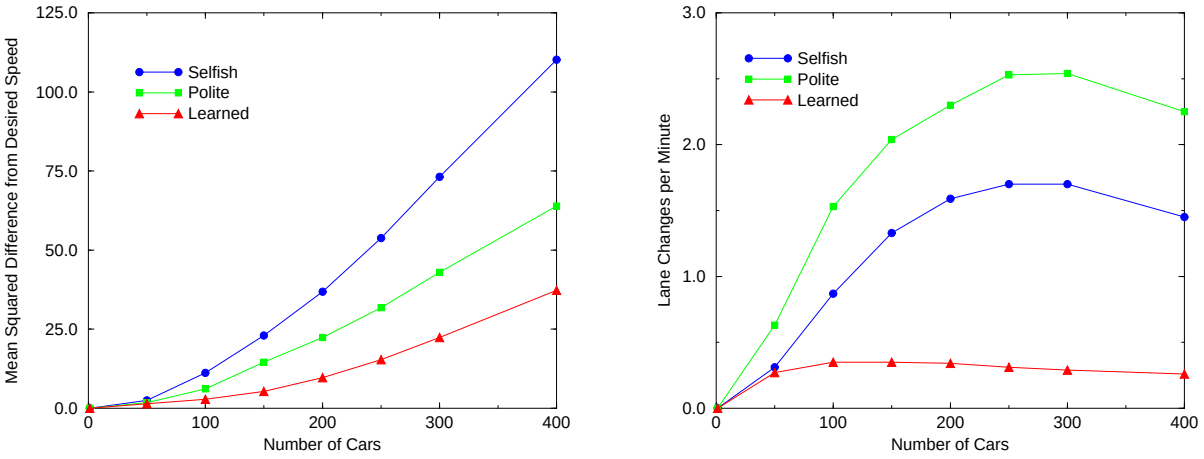


Figure 5: The performance of traffic using different lane selection strategies under different traffic densities. Figure 5(a) plots the mean squared difference between actual speeds and desired speeds from equation 1. Figure 5(b) plots the average number of lane changes per minute.

trial, the percentage of smart cars is selected randomly from a uniform distribution, with a minimum of 5 percent and a maximum of 100 percent. All other cars follow the selfish lane selection strategy outlined in section 2.1.

To simulate congestion caused by lane closures and merging, portions of either the far right or far left lanes are blocked during training. Lane closures last for one mile and exactly one closure exists at any given time. There is an equal probability that the far right or far left lane will be blocked. A lane-selection strategy perceives a blocked lane as a car with a speed of zero.

Each training run begins with a population of 75 random lane selection strategies and 25 seeded strategies, which are modified by SANE and the local learning module over 30 simulated driving hours. SANE keeps track of the best strategy found so far based on its performance over a trial. When a better strategy is found, it is saved to a file for later testing. The saved strategies are each tested over ten 2000-second trials and the best is considered the final strategy of the experiment.

4.2 Evaluation of Intelligent Lane Selection

The first experiments were designed to evaluate the merits of intelligent lane selection. Here we are not interested in the aggregate performance over several learning runs, but rather in the performance of a single strategy that could be used in traffic. Thus, the tests in the this section were conducted on the best learned strategy found over five training runs. Experiments in the next section evaluate the learning system and provide learning curves that are averaged over all training runs.

4.2.1 Experiment 1: Traffic Densities

The first study compares the performance of traffic under different traffic densities using three different lane-selection schemes: a selfish strategy, a polite strategy, and the learned strategy. The selfish and polite strategies operate as described in section 3.2. The learned strategy is the best strategy (according to fitness over the ten test trials) from the five training runs. Strategies were tested over car densities of 50 to 400 cars per 13.3 miles and performance was measured

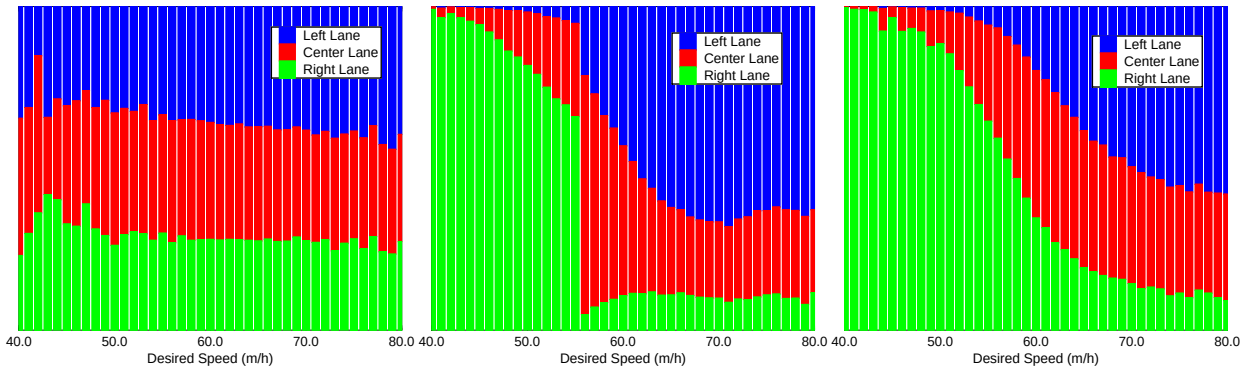


Figure 6: Utility of lanes with respect to desired speeds for the (a) selfish, (b) polite, and (c) learned strategies. The graph shows the percentage of time that cars drive in the left, center, and middle lanes as a function of desired speeds. These tests used a traffic density of 200 cars per 13.3 miles.

over 20 simulations at each density. In this experiment, all cars on the highway employed the same strategy and there were no lane closures. Since learning only occurred using 200 cars, this experiment examines generalization of the learned strategy to sparse and dense traffic.

Figure 5(a) shows the error in driving speed of the selfish, polite, and learned strategy under different traffic densities. The error is computed from the first term in equation 1 and represents the average squared difference between actual speeds and desired speeds in m/h^2. The figure shows the clear advantage of the learned strategy. In sparse traffic (50-100 cars), the performance of the three strategies is comparable; however, in more dense traffic, the learned strategy produces significantly lower divergence from desired speeds. At a density of 200 cars, the learned strategy incurs only a quarter of the error of the selfish strategy and less than half the error of the polite strategy. The selfish strategy error grows much faster in dense traffic than the polite and learned strategies, because of the many bottlenecks generated by the unyielding, slow drivers. The polite strategy solves many of these bottlenecks by moving slower drivers to the right, but still maintains a squared error of at least 20 m/h^2 over the learned strategy.

Figure 5(b) plots the average number of lane changes for each car under the three selection strategies. There is a large contrast in the lane-changing behaviors between the polite and learned strategies. Even in very sparse traffic, the polite strategy produces over twice as many lane changes as the learned strategies. In heavy traffic, the polite strategy calls for almost nine times as many lane changes. The learned strategies reach a maximum lane change rate of 0.35 changes per minute, whereas the polite strategy produces reaches 1.53 lane changes per minute. The selfish strategy generates fewer lane changes than the polite strategy, since it does not have a yielding component; however, it still generates over five times as many lane changes as the learned strategy in denser traffic. Thus, compared to both the selfish and polite strategy, the learned strategy makes far fewer lane maneuvers, which should increase driver acceptance of intelligent lane selection and hopefully reduce accident rates.

Figure 6 provides a visualization of lane utilization under the different selection strategies for a density of 200 cars. Each graph represents an average over twenty simulations of the percentage of time a driver with a given desired speed spends in each lane. The selfish strategy, shown in Figure 6(a), assigns no lane bias to faster or slower drivers, and thus drivers at different speeds are

Table 1: The distribution of traffic for the three lane selection strategies.

	Left Lane	Center Lane	Right Lane
Selfish	0.35	0.35	0.30
Polite	0.35	0.26	0.39
Learned	0.25	0.27	0.48

spread across all three lanes fairly evenly. The polite strategy, in Figure 6(b), does bias slow drivers towards the right lane and fast drivers towards the left lane, but does so with a rigid partition at 55 m/h. Thus, a car with a desired speed of 54 m/h behaves quite differently than a car with a desired speed of 56 m/h. This partition comes from the polite rule that moves cars traveling slower than 55 m/h to the right lane. The learned strategy, in Figure 6(c), produces a much smoother lane utilization bias. The slowest cars travel primarily in the right lane and, as desired speeds rise, the utilization of the middle and left lanes steadily increase.

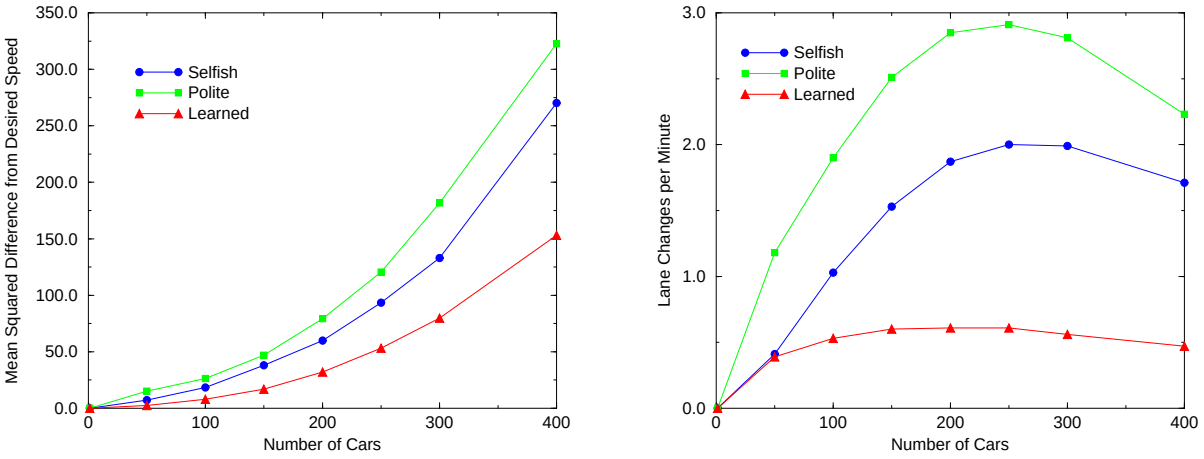
Another contrast between the three strategies lies in the overall utilization of the three lanes across all speeds. Table 1 shows the overall lane distribution for all cars. The learned strategy has a significant bias towards the right lane and places almost half of the cars there. This organization seems reasonable and quite effective since slower cars encounter fewer slower preceding cars and should operate efficiently in higher traffic density than faster cars. The learned lane-selection strategy essentially moves half of the traffic to the right lane and uses the middle and left lanes to organize the faster traffic. It is also important to note from Figure 6 that the faster cars do appear in the right lane, but the slower cars never appear in the left lane. The likely reasoning is that a slow car in the left lane causes large disruptions to traffic flow, whereas a fast car in the right lane will normally only disrupt its own performance.

4.2.2 Experiment 2: Lane Closures

The next test evaluated the three strategies in the presence of lane closures. Recall that during training, one mile of either the far left or far right lane was closed and the location of the closure changed every 500 simulated seconds. Lane closures were replicated in testing to evaluate each strategy's ability to handle high congestion areas created by merging traffic. The degree of merging in these tests is extreme (a blocked lane every 13 miles) to test the robustness of the three strategies.

Figure 7 plots the mean squared error in desired speeds and the average number of lane changes with closed lanes. Surprisingly, the polite strategy performed very poorly when portions of lanes were blocked. Figure 7(a) shows that under a high degree of merging, it is better to act greedily than politely. The large errors that the polite strategy incurs come when portions of the far right are closed. Since the polite strategy directs all of its slow drivers into the right lane, it becomes very difficult to merge them back into the two faster lanes when the right lane is blocked. This difficulty causes large bottlenecks in the right lane and creates very high errors in desired speed. Since the selfish strategy assigns no lane bias based on driving speed, it is not as affected when the right lane is closed.

Although the learned strategy also directs its slower drivers to the right lane, it is not as affected as the polite strategy to bottlenecks caused by right lane closures. Under the learned strategy, faster drivers in the center and left lanes maneuver to allow slower drivers to more easily merge, which eases congestion. These seemingly altruistic behaviors were learned because reinforcement is given



(a) (b)

Figure 7: Traffic performance when portions of lanes are blocked.

based on the aggregate performance of traffic. Additionally, the learned cars have relative speed sensors that can detect slow speeds in preceding traffic. Thus, the learned strategy can merge the cars much earlier than the polite strategy, which does not begin to merge until a preceding car or lane block forces it to slow down. As in the previous experiment, the learned strategy incurs substantially lower driving errors and performs only a fraction of the lane change maneuvers as the other two strategies.

4.2.3 Experiment 3: Four Lanes of Traffic

As noted in section 2.2, we designed the controller architecture to be independent of the number of highway lanes. The input does not denote the actual lane the car is in, and the output only reflects whether the left or right lane is better than the current lane. Thus, in principle an effective strategy formed on a three-lane highway should perform well on a four-lane highway. Our third experiment tests this hypothesis by expanding the highway capacity to four lanes. Since the learning system only experienced three-lane highways in training, this experiment also serves as another test of generalization in the learned strategy.

Figure 8 plots the error in driving speed and average number of lane changes using four lanes of traffic. Since there is more lane capacity, up to 600 cars were used in this study. The figure shows that the learned strategy achieves the same performance gain over the polite and selfish strategy in four lanes of traffic as it did in three lanes. In dense traffic, the learned strategy incurs one third to one quarter of the driving speed error of the selfish strategy and one half of the error of the polite strategy. As with three lanes of traffic, the polite and selfish strategy make substantially more lane change maneuvers than the learned strategy.

Figure 9 provides a visualization of the lane utilization of the three strategies with four traffic lanes using 300 cars. As in Figure 6, the selfish strategy assigns no lane bias based on driving speed, and the polite strategy exhibits a sharp transition between slow and fast driving styles. The graph of the learned strategy is also very similar to Figure 6(c), which demonstrates that the strategy does indeed generalize to more than three lanes.

The learned strategy continues to encumber the right lane with many slow drivers and use the other lanes to organize the faster drivers. Under four lanes of traffic, however, the fastest drivers are

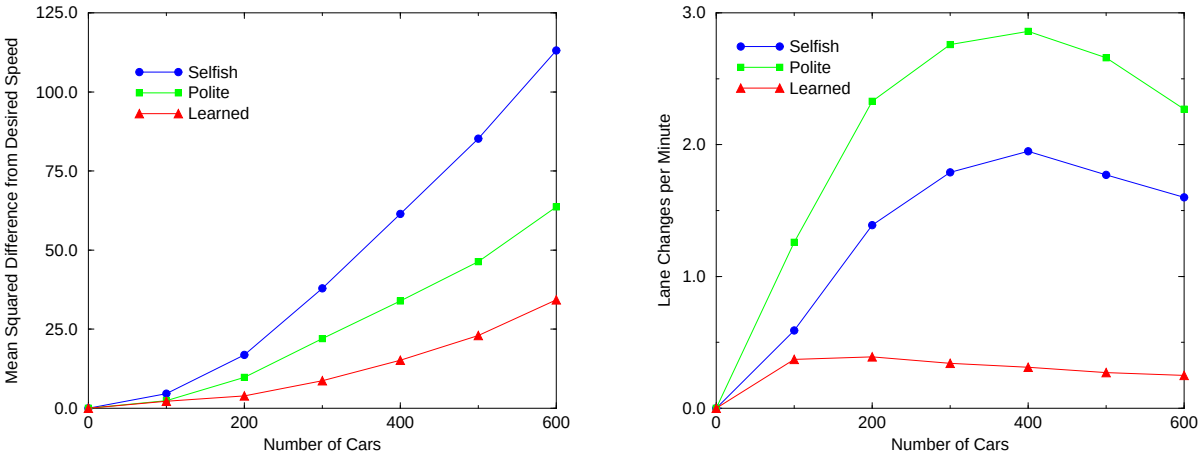


Figure 8: Traffic performance with four driving lanes. No lanes were blocked in this test.

placed more consistently in the left lane than under three lanes. For example, drivers with desired speeds of 80 m/h drive in the left lane 83% of the time with four lanes traffic, compared to 57% of the time with three lanes. It seems that most of the traffic organization occurs in the middle two lanes, with the average-speed drivers. This strategy is reasonable, since the average-speed drivers make two different types of lane changes: passing slow cars and yielding to fast cars, and therefore must reorganize more frequently.

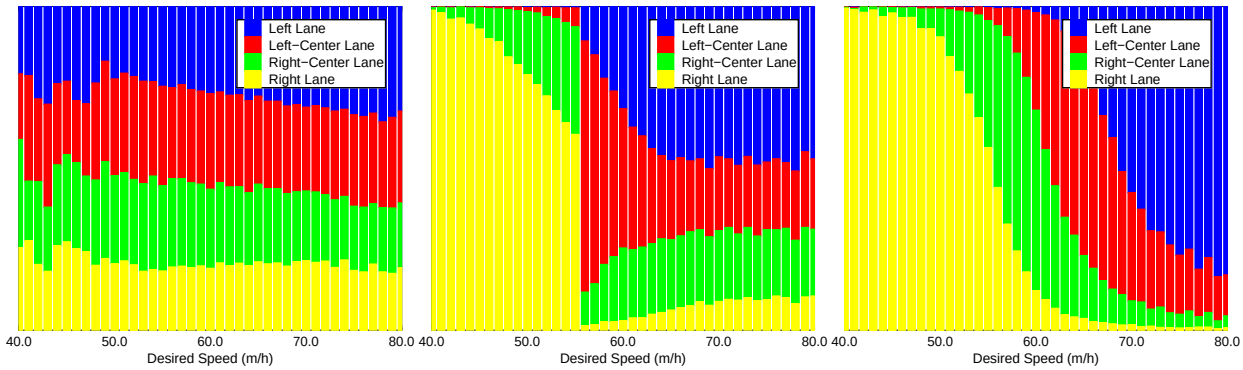
4.2.4 Experiment 4: Mixing Selfish and Learned Strategies

The fourth experiment evaluated the learned strategy in the presence of selfish cars. The aim was to examine the robustness of the smart car's group behavior to cars that do not follow the same rules of behavior. We were interested in how quickly the learned strategy breaks down as more selfish drivers are added and how many smart cars are necessary to make cooperative behavior worthwhile.

Figure 10 shows the error in driving speeds under different smart car distributions with and without lane closures. The figure plots the speed error for both the smart cars and the selfish cars, and it illustrates how the performance for both improves with the number of smart cars. The two graphs show that, even with as few as 5% smart cars, there is incentive to cooperate. At 100% selfish traffic with no lane closures, cars average a 36.80 m/h^2 driving error, while at 95% selfish traffic the error drops to 34.40 m/h^2. Although this improvement in not substantial, it demonstrates that very few smart cars are necessary to improve the overall traffic behavior.[1] Moreover, performance improves steadily as more cars cooperate, which provides further motivation to drive cooperatively. Finally, at 100% smart cars the average speed error drops to 9.66 m/h^2, which is approximately one fourth of the error when all traffic is selfish. The performance increase is not as great when portions of lanes are blocked, but there remains a steady improvement as more smart cars are added.

We should note that, since our goal was to motivate cooperative driving, we trained the smart cars only to optimize the performance of other smart cars. In other words, during the learning process equation 1 is only evaluated over the smart cars. Thus, the reduction of speed error in

[1]The one exception occurs with 20 percent smart cars. Additional runs confirmed that the two controllers had the same error at this level, but we have not yet explained the effect.



(a) (b) (c)

Figure 9: Utility of lanes with respect to desired speeds for the (a) selfish, (b) polite, and (c) learned strategies with four traffic lanes.

the selfish cars is a "side effect" of the cooperative behavior of the smart cars. This effect is understandable, since the selfish cars can take advantage of yielding smart cars, but they do not benefit more than the smart cars themselves.

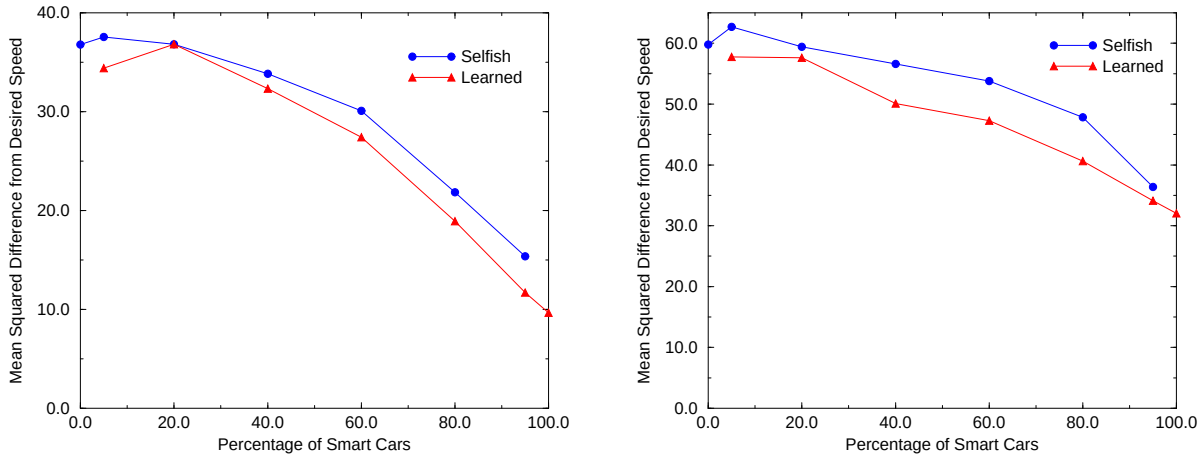
Figure 11 plots the average number of lane changes per minute for both the selfish and smart cars. Unlike the speed error plots, there is a large contrast in the selfish and smart performances. The smart cars maintain an average of less than 0.5 lane changes per minute across all distributions. The selfish cars, however, change lanes at the rate of 1.87 per minute when no smart cars are present and only go below 1.0 with 95% smart cars under no lane closures. At 95% smart cars, there are so many smart cars clearing paths for the selfish cars that lane changes are not as necessary. The lane change disparity between selfish and smart drivers provides an even stronger motivation for cooperative driving.

4.3 Evaluation of the Learning Modules

The previous experiments demonstrated how an intelligent lane-selection strategy can improve global traffic performance. The next experiment targeted the learning system and evaluated its behavior over several training runs. The goal of this experiment was to produce a performance curve as a function of simulation time and to measure the contributions of each of the different learning modules.

We tested four variations of the learning system: SANE, SANE with the local learning module, SANE with the population seeder, and SANE with both local learning and population seeding. Each variation was tested over ten different training runs in the traffic simulator with no blocked lanes. To reduce the overall CPU time for each experiment, only 50 cars were used over a 3.3 mile repeating freeway; however, the car-density of 15.1 cars/mile is identical to the previous simulations.

Figure 12 plots the learning curves for the four learning variations. Performance is measured over a 20 trial test set of which 10 trials used 100% smart cars and 10 trials used a random percentage of smart cars. The performance metric comes from a combination of experiments 1 and 4 in the previous section. The metric provides a single benchmark that evaluates strategies over situations when all cars are cooperating and when greedy cars are present. Thus, performance



(a) (b)

Figure 10: The average squared error of smart and selfish cars under different smart car distributions with (a) no blocked lanes and (b) blocked lanes.

numbers in Figure 12 are not directly comparable to any single experiment in the previous section, but they are comparable to the combined performances in experiments 1 and 4. Each curve plots the best performance found at or before the current simulation time and is averaged over the ten training runs.

The graph shows a clear improvement when either the local learning or population seeding modules are added, and the best performance is achieved when both are present. In particular, population seeding provided very effective initial strategies for the system to work with. Without seeding, early strategies achieved an average error in driving speed of around 75, whereas with seeding the error was only 32. Local learning proved beneficial whether seeding was done or not.

The learning curve also illustrates how quickly our learning approach forms good strategies. Within only three simulated hours, the system produces strategies that generate less than a 20.0 mean squared driving error from desired speeds. The weakness of the learning method, however, appears to be refinement of the best strategies. Although good strategies are found quickly, improvements to those strategies take considerably longer. For example, while the system requires only three simulated hours to reach the error level of 20.0, it requires 18 simulated hours of driving to reduce the error to 15.0.

Experience in other domains such as robotics indicates that SANE is very good at finding good areas of the solution space quickly, but that it can have trouble pinpointing the best solutions within that space (Moriarty & Miikkulainen, 1996b). The local learning module does aid in this refinement, but there remains a quickly diminishing return as simulation continues. We are currently looking into other mechanisms for solution refinement, including the addition of temporal difference learning.

5 Related Work

Intelligent lane selection appears to be a novel approach to traffic management that has received almost no attention in the traffic management literature. After a lengthy literature search and several email inquiries, we have found only one project with similar goals. Carrara and Morello

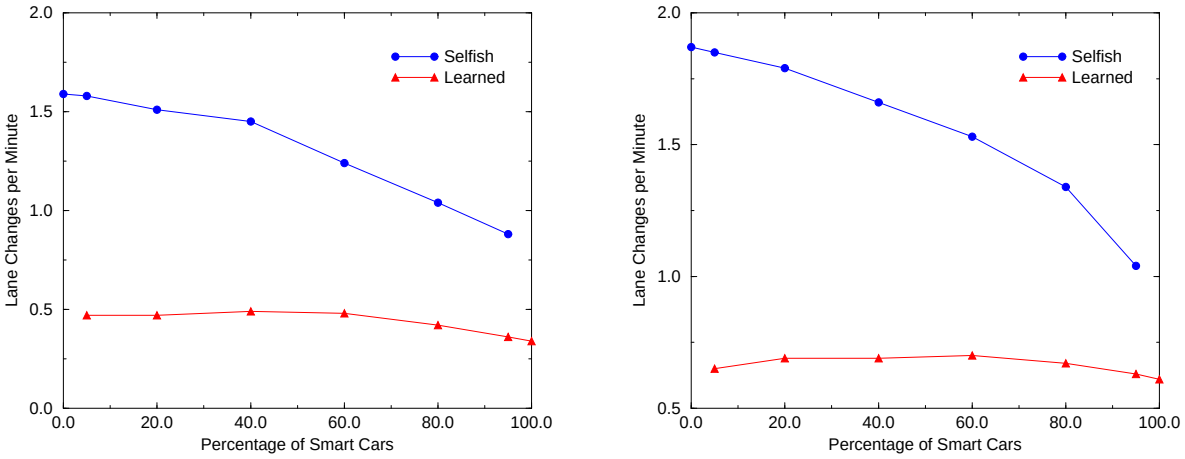


Figure 11: The average number of lane changes per minute under different smart car distributions using (a) unblocked and (b) blocked lanes.

have proposed a system called DOMINC that employs cooperative driving techniques to increase traffic efficiency.[2] The main objective of the DOMINC project is

> to explore the possible improvements and quantify benefits in traffic efficiency, comfort and safety offered by the new concept of cooperative driving and define a new RTI system for the motorway.

Carrara and Morello assume a strong communication link where each car maintains significant state information of other cars. The targeted areas of control for the DOMINC system include speed, lane changing, and inter-car distance, with the potential benefits of reduced congestion, increased safety, and higher level of driver comfort. The vision of the DOMINC project is thus very close to our formulation of car-centered traffic management. However, the paper that we have only describes the potential benefits and does not propose a specific methodology for cooperative driving. Also, it does not mention machine learning as an effective means to generate the cooperative driving strategies.[3]

There are other lane-selection systems that do not manage traffic, but that are still related to our work. For example, McCallum (1996) used reinforcement learning to train a driving agent to weave around slower and faster traffic in a task that he calls "New York driving". Here the aim is a selfish strategy for lane selection that benefits the driver's own performance. This contrasts sharply with our task, in which multiple agents that learn cooperative strategies that aim to benefit overall traffic performance. Another difference lies in the motivation of the research. McCallum created his task to exhibit the advantages of his approach to reinforcement learning, and he makes no claims that his learned behaviors will be useful in real traffic. We designed our lane-selection task to demonstrate the advantages of cooperative driving, and we do predict that our learned strategies will prove useful in real traffic. However, the two efforts share the assumption that acquiring traffic controllers is best formulated as a problem of learning from delayed reward in a

[2]The paper that we have does not include a full reference.

[3]We have been unsuccessful in our attempts to find out more about the DOMINC project, and would appreciate any help in that regard.

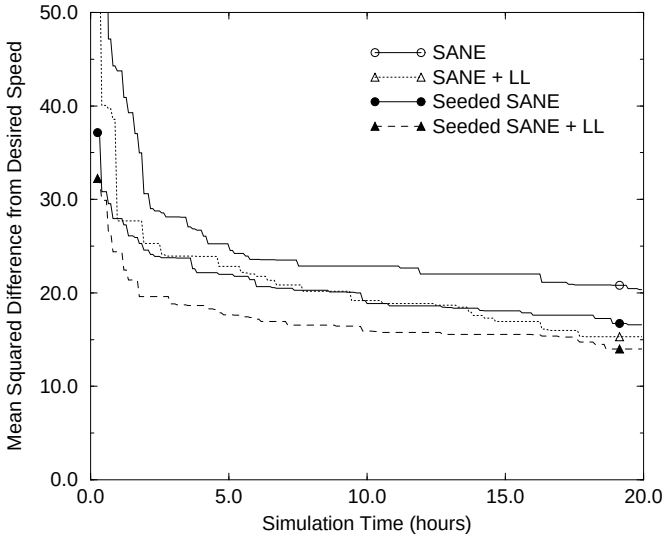


Figure 12: The average performance of the system as a function of simulation time. The graph shows four variants of the complete learning system: SANE only, SANE with local learning, SANE with population seeding, and SANE with both local learning and population seeding.

driving environment, rather than learning from traces of correct behavior. McCallum's learning method relies on a temporal-difference strategy, but he could instead have used an evolutionary approach, just as we could have employed his method for our distributed learning task.

Sukthankar, Baluja, and Hancock (1997) used an approach similar to an evolutionary algorithm to form a voting scheme that makes tactical driving decisions. Their system learned controllers for a single car that managed both lane selections and driving speed to maximize desired speed, avoid collisions, and exit the highway. Because the authors were concerned with the behavior of a single car, rather than overall traffic behavior, they used a different fitness function that, most likely, was biased toward selfish controllers that do not take into account the performance of other cars. The key difference is that we view lane selection as a multi-agent problem, whereas Sukthankar et al. view it as a single-agent problem. However, there are enough common aims that we could adapt their learning technique to our task and vice versa.

The Bayesian Automated Taxi (BAT) project aims to build a fully automated vehicle that can drive in normal traffic (Forbes, Huang, Kanazawa, & Russell, 1995). The system includes a hierarchy of control modules that range from trip planning to automated driving, including a module for lane selection at the middle level. However, the current system makes lane changes only to maintain the target speed of the individual car. We are interested to see whether future work on BAT will emphasize selfish or cooperative driving. The project does address an important issue – noise or inaccuracies in sensory data – that we have not. In line with our vision, Forbes et al. hold that smart cars will initially coexist with greedy, human-driven cars. However, whereas we currently assume that smart cars receive perfect information about other cars' speeds and positions, BAT treats these values as uncertain and uses a Bayesian network to maintain to estimate other drivers' speeds and intentions. Learning occurs by adjusting the conditional probabilities based on actual sensor readings. Robustness to sensor noise is an important attribute, and we hope to incorporate similar ideas in our future work.

Research on advanced traffic management *has* focused on issues of larger-scale traffic control. One popular method for lane selection in automated freeways determines a fixed speed for each lane and then assigns each driver to a lane based on the length of their trip (Hall, 1995; Ramaswamy et al., 1997). This approach encounters several problems within our formulation of the traffic management task. First, it assumes that all drivers will follow their instructions, and it remains unclear how this management strategy will perform if selfish drivers ignore their lane assignments. Our approach, on the other hand, has proven to be robust in the presence of selfish drivers (figure10). Second, it forces many drivers to go faster than their desired speeds, and we predict that slower drivers will find such instructions disagreeable. Nevertheless, it may be possible to design variations on this scheme that better meet our aims, so we should not rule out the basic framework without additional thought.

The multi-agent framework used in our experiments is similar to the approach proposed by Schmidhuber (1996) for learning cooperative behaviors through reinforcement learning. As in Schmidhuber's method, each agent does not attempt to model the behavior of other agents, but rather treats their features as characteristics of the environment. Control policies are formed that use sensors to detect characteristics of other agents, such as location and speed, to generate effective control decisions. Schmidhuber demonstrated that agents can achieve cooperative behavior without modelling each other, which is consistent in the results of our own simulations.

6 Discussion and Future Work

The experimental results in this paper are encouraging and demonstrate the potential benefits of intelligent lane selection. Smart cars significantly reduced both the difference between actual and desired speeds and the number of lane changes. However, as described in section 4.1, there were several assumptions in the current simulation model that we hope to address in the near future.

One of the assumptions that could affect performance is the highway architecture. On-ramps and off-ramps introduce interesting dynamics into the traffic environment that will certainly affect lane selections. A car that is taking an immediate right off-ramp should not move left even if a preceding car causes it to reduce its speed. Also, a car entering the highway with a high desired speed should not immediately move to the left lane, since it will need to accelerate to that speed. Our smart cars do appear robust to external factors that influence lane selections, such as selfish cars, and should adapt to fit their particular environment. If the environment contains off-ramps and on-ramps, and if cars encounter them during training, then control policies should evolve that account for them.

Another problem is that the number of lane changes we observed does not appear to reflect the actual behavior of people. Drivers have specific speed tolerances that influence when they change lanes. For example, a driver may change lanes only if his speed drops 10 m/h below his desired speed and if there is very little traffic in another lane. Since these tolerances are difficult to obtain and quantify, our driver models assume that a lane change is attempted as soon as the speed drops below the desired speed. We have also assumed that the desired speed for each driver is constant, whereas actual drivers may change this quantity over the course of a trip. In the future, we hope to demonstrate that our approach still works for more realistic models of driver behavior.

In this paper, we presented intelligent lane selection as an approach to letting drivers approximate their desired speeds and to minimizing the number of lane changes. However, lane selection could attempt to optimize other performance criteria as well. Examples include selecting lanes

to increase overall traffic throughput, selecting alternative routes to balance traffic loads over a highway system, or letting emergency vehicles easily pass through to their destination. We believe these criteria can be easily incorporated into equation 1 and thus improved through reinforcement learning.

Another area where cars can coordinate behavior to maximize global utility is speed control. For example, cars can vary speeds to let other cars easily merge onto the highway and avoid unnecessary bottlenecks. Reinforcement learning could generate local speed control policies that generate target speed based on information about the preceding traffic and the driver's current speed. Collectively, the speed control policies should promote overall traffic throughput and reduce areas of congestion.

7 Summary and Conclusions

Coordination of local car behaviors is a novel approach to traffic management that poses a challenging problem to both artificial intelligence and machine learning. In this paper, we proposed one formulation of this problem: intelligent lane selection to maintain desired driving speeds and reduce lane changes. Given only information on the local traffic patterns and the desired speed, cars can coordinate local lane changes to let faster traffic pass through while still allowing slower traffic to maintain desired speeds.

We described and evaluated an approach that uses supervised and reinforcement learning to generate the lane-selection strategies through trial and error interactions with the traffic environment. Compared to both a selfish strategy and the standard "yield to the right" strategy, the smart cars maintained speeds closer to the desired speeds of their drivers while making fewer lane changes. Additionally, intelligent lane selection was shown robust in the presence of selfish drivers. Traffic performance improves even when as few as five percent of the cars cooperate. Future work will explore more realistic traffic and driver models, as well as variations on the task of coordinating driver behaviors.

References

Carrara, M., & Morello, E. Advanced control strategies and methods for motorway of the future. In *The Drive Project DOMINC: New Concepts and Research Under Way.*

Dietterich, T. G. (1990). Exploratory research in machine learning. *Machine Learning*, *5*, 5–9.

Eskafi, F. (1996). *Modeling and Simulation of the Automated Highway System.* Ph.D. thesis, Department of EECS, The University of California at Berkeley.

Forbes, J., Huang, T., Kanazawa, K., & Russell, S. (1995). The BATmobile: Toards a bayesian automated taxi. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)* Montreal, CA.

Gilmore, J. F., Elibiary, K. J., & Forbes, H. C. (1994). Knowledge-based advanced traffic management system. In *Proceedings of IVHS America* Atlanta, GA.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA.

Grefenstette, J. J., Ramsey, C. L., & Schultz, A. C. (1990). Learning sequential decision rules using simulation models and competition. *Machine Learning*, *5*, 355–381.

Hall, R. W. (1995). Longitudinal and lateral throughput on an idealized highway. *Transportation Science*, *29*, 118–127.

Holland, J. H., & Reitman, J. S. (1978). Cognitive systems based on adaptive algorithms. In *Pattern-directed Inference Systems*. Academic Press, New York.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

Kagolanu, K., Fink, R., Smartt, H., Powell, R., & Larson, E. (1995). An intelligent traffic controller. In *Proceedings of the Second World Congress on Intelligent Transport Systems*, pp. 259–264 Yokohama, Japan.

McCallum, A. K. (1996). Learning to use selective attention and short-term memory in sequential tasks. In *Proceedings of Fourth International Conference on Simulation of Adaptive Behavior*, pp. 315–324 Cape Cod, MA.

Moriarty, D. E. (1997). *Symbiotic Evolution of Neural Networks in Sequential Decision Tasks*. Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin.

Moriarty, D. E., & Miikkulainen, R. (1996a). Efficient reinforcement learning through symbiotic evolution. *Machine Learning*, *22*, 11–32.

Moriarty, D. E., & Miikkulainen, R. (1996b). Evolving obstacle avoidance behavior in a robot arm. In *From Animals to Animats: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (SAB-96)*, pp. 468–475 Cape Cod, MA.

Pomerleau, D. (1995). Ralph: Rapidly adapting lateral position handler. In *Proceedings of the 1995 IEEE Symposium on Intelligent Vehicles* Detroit, MI.

Pomerleau, D. A. (1992). *Neural Network Perception for Mobile Robot Guidance*. Ph.D. thesis, Carnegie Mellon University.

Pooran, F. J., Tarnoff, P. J., & Kalaputapu, R. (1996). RT-TRACS: Development of the real-time control logic. In *Proceedings of the 1996 Annual Meeting of ITS America*, pp. 422–430 Houston, Tx.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106.

Ramaswamy, D., Medanic, J. V., Perkins, W. R., & Benekohal, R. F. (1997). Lane assignment on automated highway systems. *IEEE Transactions on Vehicular Technology*, *46*, 755–769.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E., & McClelland, J. L. (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pp. 318–362. MIT Press, Cambridge, MA.

Sammut, C., Hurst, S., Kedzier, D., & Michie, D. (1992). Learning to fly. In *Machine Learning: Proceedings of the Ninth International Workshop*, pp. 385–393. Morgan Kaufmann.

Schmidhuber, J. (1996). A general method for multi-agent reinforcement learning in unrestricted environments. In *Adaptation, Coevolution, and Learning: Papers from the AAAI Spring Symposium*, pp. 84–97. AAAI Press.

Sukthankar, R., Baluja, S., & Hancock, J. (1997). Evolving an intelligent vehicle for tactical reasoning in traffic. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Sutton, R. S. (1988). Learning to predict by the methods of temproal differences. *Machine Learning*, *3*, 9–44.

Varaiya, P. (1993). Smart cars on smart roads: Problems ofcontrol. *IEEE Transactions on Automatic Control*, *38*, 195–207.

Varaiya, P., & Shladover, S. (1991). Sketch of an ivhs systems architecture. Tech. rep. PATH Research Report UCB-ITS-PRR-91-03, Instution for Transportation Studies, The University of California at Berkeley.

Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3), 279–292.

Whitley, D., Dominic, S., Das, R., & Anderson, C. W. (1993). Genetic reinforcement learning for neurocontrol problems. *Machine Learning*, *13*, 259–284.

Wilson, S. W. (1994). ZCS: A zeroth level classifier system. *Evolutionary Computation*, *2*(1), 1–18.