Approximation strategies for routing in stochastic dynamic networks

Tomáš Šingliar
Department of Computer Science
University of Pittsburgh
tomas@cs.pitt.edu

Miloš Hauskrecht
Department of Computer Science
University of Pittsburgh
milos@cs.pitt.edu

Abstract

In this work, we study a special semi-Markov decision process that formalizes a route-planning problem in stochastic transportation networks. We explore two versions of the planning problem: one in which the planner knows the initial traffic situation but does not have access to further information once it begins executing the plan (open-loop), and one in which the planner receives continuous traffic updates (closed-loop). Since exact versions of these planning problems are intractable, we study their heuristic approximations. We present a new class of Monte-Carlo route planning algorithms that optimize the route using a sample of multiple traffic-state trajectories. We show that these methods are able to outperform Monte-Carlo methods based on greedy policy look-aheads applied most frequently to solve the stochastic decision problems.

1 Introduction

The dynamic behavior of real-world traffic systems is subject to stochastic fluctuations. It is the result of complex spatiotemporal interaction between traffic volumes, speeds and physical infrastructure properties. The stochastic behavior of the systems has a huge effect on a variety of traffic-related tasks. In this paper, we study the route planning problem in which the goal is to guide a vehicle through traffic so that the target destination is reached in the minimal expected time.

Multiple versions of the route-planning problem (Pallottino & Scutella 2003; Bander & White 2002; Fu, Sun, & Rilett 2006) can be formulated depending on the optimization goal and the available information. In our work, we assume that the information about the traffic situation is available to the planner before and during the plan execution. The collection of online traffic information is made possible by sensor networks installed on many highways. We assume that the routing decisions made for the individual target vehicle have a negligible effect on overall traffic volumes and the future traffic behavior.

In general, the vehicle routing problem can be formulated as a Semi-Markov decision process (SMDP), where states correspond to the traffic conditions (speeds, volumes in different parts of the road network) and the current location of the vehicle we want to route; and the actions correspond to possible route choices the driver can make at intersections. A dynamic model represents the behavior of the traffic through time. The costs correspond to the time it takes the vehicle to travel to the next location.

A traffic system involves a large number of interconnected road components. This leads to a semi-MDP with a high-dimensional state space. In addition, all quantities (volumes, speeds etc) are continuous which makes the exact solution of the full semi-MDP infeasible. To alleviate the state space problem, we study a variety of route planning approximations.

Many SMDP-solving approaches have been developed. Among the most popular are reinforcement learning (RL) algorithms: (Bradtke & Duff 1995) bring discrete-time MDP into the semi-MDP context. RL approaches that derive from single-step, $TD(0)$-type updating (Das *et al.* 1999) are likely to suffer poor initial performance in problems with delayed rewards. More importantly, all work with an implicit confidence that the future state of the environment, and therefore the action costs, can be described by clearly defined—if complex—probability distribution. This paper acknowledges that it is not known well enough how to predict the state and therefore cost of future actions in transportation networks that operate near their capacity (Nagel & Rasmussen 1994). In principle, such model of the future can be built in the process of solving the SMDP with methods such as Adaptive RTDP (Bradtke 1994) or any of the above methods but convergence would almost certainly be slow. A contribution of this paper is the use of an underlying flow-dynamic model in a decision context. Use of a dynamic model should allow the routing algorithm to produce usable routes out-of-the-box, without lengthy learning periods, thus perhaps opening the door to subsequent (reinforcement) learning steps that refine it.

We develop a new Monte Carlo method that generates a small sample of traffic state trajectories[1]

and uses an A^*-like algorithm to obtain the optimal route for each trajectory. The routes found are then compared via a second round of Monte Carlo evaluation and the best route is used to approximate the optimal routing decision. The

[1]To avoid confusion early on, by state-trajectory we mean time-indexed sequences of environment states, as opposed to a sequence of locations to move through in the physical space. We use the term "route" for the latter.

advantage of the method is that it lets us evaluate the longer-term effects of routing decisions, not just its first few steps as is typically done in greedy look-ahead methods (Bertsekas 1995).

To obtain initial insight into the performance of our algorithms, we test them on the routing problem in a traffic network with 144 (unidirectional) road segments and 61 intersections. We show that our new Monte Carlo method outperforms other route planning methods on problems in which information about the current traffic state is provided either one-time at the beginning of or continuously during the plan execution.

2 The model

The vehicle routing problem can be formulated as a special semi-Markov decision process (MDP) (Howard 1963; Jewell 1963; Puterman 1994). The state of the process is represented by the traffic state component $\mathbf{S}$ and the vehicle location component X. A traffic state, $\mathbf{s} \in S$, is defined as $\mathbf{s} = \{r_1, v_1, \cdots r_i, v_i \cdots, r_N, v_N\}$ where (r_i, v_i) values represent the traffic speed and the traffic volume on the i-th road segment. The traffic volume v_i is the number of vehicles traveling segment i. The location component x represents the current location of the vehicle. We assume the domain of X consists only of endpoints of road segments.

Actions of the semi-MDP correspond to possible route choices the vehicle can take at road intersections. The intersections are uniquely defined by the location component of the state. For every location x there is a finite set of route actions $A(x)$ that is independent of time.

2.1 Model of network dynamics

The dynamics of the system represents the behavior of the traffic system in time. It is fully described by the conditional density:

$$p(\mathbf{s}', x', \Delta t|\mathbf{s}, x, a) \tag{1}$$

where s and s' represent the current and the next traffic state, x and x' are the current and the next locations of the vehicle, a is the routing action taken at x, and Δt is the time of the transition, that is, the time it takes to get from x to x' under the traffic condition s.

Traffic state model. To define our model we assume that actions of our driver are negligible for the overall evolution of traffic. This lets us model traffic state evolution independent of routing decisions. In particular, we define a traffic component of the stochastic process using the state transition distribution $p(\mathbf{s}^*|\mathbf{s})$. We assume all transitions occur at fixed time intervals of length δt. Our model of $p(\mathbf{s}^*|\mathbf{s})$ is inspired by continuum flow laws, a common basis for macroscopic traffic flow models such as METANET (Kotsialos *et al.* 2002). First the conditional probability decomposes along individual road segments:

$$p(\mathbf{s}^*|\mathbf{s}) = \prod_{i=1}^{N} p(r_i^*, v_i^*|\mathbf{s})$$

where r_i^* and v_i^* are speeds and volumes for the segment i in the next step (after time δt). Next, the volume-speed relation decomposes as $p(r_i^*|v_i^*)p(v_i^*|\mathbf{s})$. This reflects our view of volumes as the primary state component for traffic interactions among segments. Speeds are thought of as a "dependent variable", through the volume-speed relationship $p(r_i^*|v_i^*)$ described below. The $p(v_i^*|\mathbf{s})$ is modeled as a Gaussian distribution $v*_i \sim N(\bar{v}_i^*, \sigma_i^2)$ whose mean is determined by flows in the neighboring segments. Let $Pre(i)$ and $Succ(i)$ denote the predecessors and successors of i, that is the road segments that feed into and leave the i-th segment, respectively. We define the mean of the distribution to be:

$$\bar{v}_i^* = v_i + v_{i0} + \left[\sum_{j \in Pre(i)} \theta_{ji} \frac{r_j^* \delta t}{L_j} v_j\right] - \frac{v_i \delta t}{L_i} \sum_{j \in Succ(i)} \theta_{ij} r_j^* \tag{2}$$

where the term v_{0i} captures the expected inflow/outflow through the unmeasured on- and off-ramps on the i-th segment and L_i is the length of the road segment. Each θ_{ji} represents the proportion of flow on j that reaches i. [2] Equation 2 essentially expresses the flow continuity law, with first summation term being the inflow to link i while the second term represents the corresponding outflow. Note how the outflow speed is determined by the speeds at the downstream links. Through this mechanism congestions propagate upstream.

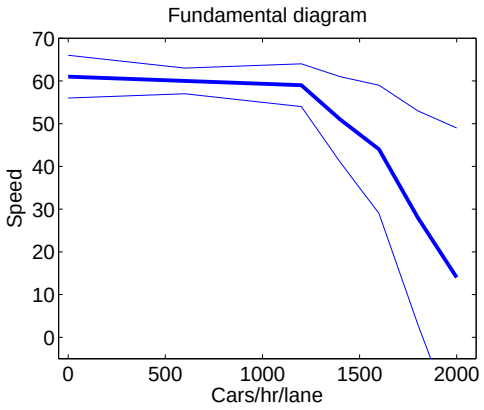
The second term of the traffic model, $p(r_i^*|v_i^*)$, derives from the so-called fundamental traffic diagram (Kerner 2003) that relates traffic volumes and speeds (Figure 1a). We approximate the conditional density $p(r_i^*|v_i^*)$ by dividing the volume range to subintervals and by using a piecewise-linear approximation that maps the volumes to their corresponding mean speeds for each subinterval. The natural variation around the mean in each subinterval is modeled by a Gaussian noise rectified to positive numbers. Figure 1a shows an example of the volume speed diagram implemented in the model.

Traffic-location model. The state of the traffic defines only one component of our semi-Markov model. We need to add a traffic-location model that ties the location and the state s. It is based on the following decomposition:

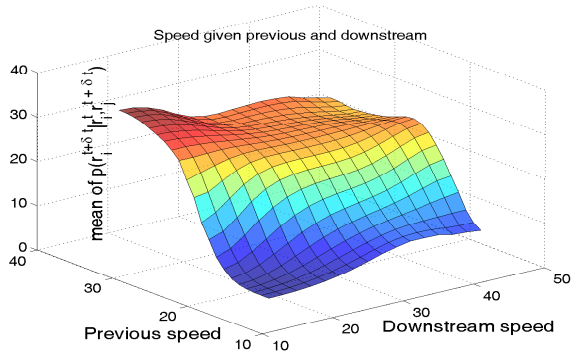
$$p(\mathbf{s}', x', \Delta t|\mathbf{s}, x, a) = p(\mathbf{s}', \Delta t|\mathbf{s}, x, a, x')p(x'|\mathbf{s}, x, a).$$

To implement individual terms of the decomposition we make the following assumptions. First, the next location x' is a deterministic function of the current location x and the road segment a chosen and is independent of the traffic state. This assumption reflects the fact that once we take a road a in location x we eventually reach the segment's end at x'. To model $p(\mathbf{s}', \Delta t|\mathbf{s}, x, a, x')$ we make use of the above traffic state model with time step δt. We assume that Δt is larger than δt. The travel time Δt for the road segment i is a function of speeds r_i of the stochastic traffic model and the segment length L_i.

[2] In this work we assume that the flow distribution parameters θ_{ij} and σ_i^2 for road segments are known. For real networks they can be estimated through regression methods (Hastie, Tibshirani, & Friedman 2001).



(a)



(b)

Figure 1: a) A fundamental diagram modeling the interaction between the volume and speed. Average and one-standard deviation contours are shown. b) The spatial and temporal dependencies: empirical mean of $p(r_i^*|r_i, r_j)$ with $i \in Pre(j)$ obtained by sampling the dynamic model with $\delta t = 5$ min.

Let $(r_i^{(0)}, r_i^{(1)}, r_i^{(2)}, \ldots)$ be a sequence of speeds on segment i. Then the segment travel time $\Delta t = A\delta t$, such that $A = \underset{K}{\operatorname{argmin}} \left[\delta t \sum_{u=1}^{K} r_i^{(u-1)} \geq L_i\right]$, where L_i is the length of the segment i. In other words, the travel time is determined by the time our vehicle can cover the length of the segment with varied segment speeds. The next state $\mathbf{s}'$ is the traffic state corresponding to index A. The travel time also defines the cost component of our semi-Markov model.

2.2 Decision problem

Our goal is to navigate through the traffic so that the best expected travel time between the start and target location is achieved. The travel time corresponding to a route $\langle x_0, x_1, x_2, \cdots x_\ell \rangle$ is obtained by summing up the travel times between the consecutive locations visited by the route. At every decision point the current traffic condition may be observed and taken into account when selecting the next route choice.

The optimal solution to the decision problem is described by the Bellman equation (Bellman & Dreyfus 1962):

$$V(\mathbf{s}, x) = \min_{a \in A(x)} E[\Delta t|\mathbf{s}, x, a] + \int_{S'} p(\mathbf{s}'|\mathbf{s}, x, a) V(\mathbf{s}', x_a) \mathrm{d}\mathbf{s}' \quad (3)$$

where $V(s, x)$ is the optimal expected time for the route starting in location x at traffic condition s and x_a denotes the intersection reached from x by taking the road segment a. Briefly, the optimal expected time for the current traffic condition s and location x is obtained by optimizing the sum of the expected time for the first section of the road corresponding to the action choice a, and the optimal expected time for the remainder of the route. In our simplified model, the relationship between s, x and Δt is functional and the expectation in the first term is unnecessary.

The space of traffic conditions is continuous. Consequently, the optimal value function and the optimal policy may not have finite support. Compact parameterizations of corresponding value functions are likely to exist only under certain restrictions placed on the form of the transition model and model cost (Boyan & Littman 2001). To alleviate the problem, and to allow for information influx during the execution of the route plan, we focus on the "on-line" version of the routing algorithms that always identifies the best immediate routing choice for the current traffic condition. The route followed and executed is then formed by the sequence of such choices.

3 Fixed-route planning

Assume first a simplified version of the routing in which the initial traffic situation s_0 is known, but no further information is provided after the plan is being executed. Hence all routing decision are made before the execution and the plan does not allow for future changes in traffic patterns.

3.1 Snapshot approximation

The most natural approximate solution to this problem is to take s_0 and assume it does not change in time. If the time scale of changes in the dynamics of the system is much larger than actual travel times, then this approximation should give close-to-optimal results. The main advantage of the approach is that it can be solved efficiently using Dijkstra's shortest path algorithm or A^*, its heuristic extension, with costs corresponding to travel times.

3.2 Building time-dependent plans

One problem of the snapshot approximation is that it ignores temporal dependencies among traffic state variables. As a result, optimal routes for fixed traffic variable values (and hence fixed travel times) may lead to suboptimal routing solutions. To incorporate the temporal aspect of the routing problem and the fact that traffic state may evolve over time on, we extend the Dijkstra shortest path (Dijkstra 1959) method to handle time dependent travel times.

Time-dependent shortest path Assume we know the trajectory of traffic states $\mathbf{s}$, beginning in state s_0. We can easily extend Dijkstra's algorithm to this case by making the cost (travel time) for the new road segment dependent on the "total elapsed time" (see Algorithm 1). The locations reached and their best total time is then kept and can be used for pruning new search tree states. Search tree pruning is based on the following monotonicity property of transportation networks:

$t_1 + time^*(x, y, t_1) < t_2 + time^*(x, y, t_2)$ whenever $t_1 < t_2$.

In other words, arriving at a midpoint earlier will never cause later arrival to target location on an optimal path from x to y.

Mean trajectory method One way we can exploit the time dependent shortest path algorithm is to use the mean trajectory method. The method generates a set of k traffic-state trajectories of duration T starting in the initial traffic state $\mathbf{s}_0$. All traffic trajectories are generated with a fixed sampling step δt. The values of traffic variables are averaged over trajectories to yield the mean volume and speed trajectories. These quantities define the travel times to be used by the time-dependent shortest path algorithm.

Best-of-k plans method The limitation of the mean trajectory method is that the plan is generated by averaging the traffic quantities for many possible traffic state evolutions. As a result, spatial and temporal interactions that occur in individual trajectories may be blurred and never properly accounted for. To alleviate the problem we propose the BEST-OF-k plans method. Similarly to the mean trajectory method, the method first generates a set of k traffic-state trajectories of duration T starting in the initial traffic state $\mathbf{s}_0$. However, instead of combining the trajectories via averaging, a separate plan is built for each trajectory using the time-dependent variant of the shortest path algorithm. This yields a set of k paths for each possible traffic state evolution. To compare and select the best of these k paths M additional state-time trajectories are sampled from the dynamic model and the average (over the M samples) travel time for each path is determined. [3] The path evaluating with the least average travel time is selected.

The advantage is that unlike the mean trajectory method, BEST-OF-k considers entire state trajectories, allowing for capture of dependent behavior of network components. Most important of such correlated behaviors is the propagation of congestion upstream which leads to nonlinear joint cost structures (travel time is a non-linear function of speed). The limitation is the computational effort spent on individual plan optimizations and their comparison through the second round of Monte Carlo evaluation.

4 Route planning with information feedback

The route optimizations discussed so far pick a single route before its execution and they do not account for traffic contingencies that may arise during the execution. As a result, the commitment to a single route may lead to a suboptimal behavior. In this section we focus on online algorithms that consider new information about traffic flow state during the execution of the plan. For simplicity, we will assume that the information arrives at decision points. Clearly, flexible planning algorithms have more up-to-date information and should dominate the fixed strategies.

k-step look-ahead greedy This online approach relies on a heuristic estimate of the value function $\hat{V}$ in each unfolding of the Bellman equation:

$$V(s, x) = \min_a \left[\frac{1}{n} \sum_{j=1}^{n} [\Delta t^{(j)}(s, x, a) + \hat{V}(s^{(j)'}, x_a)] \right],$$

where $\Delta t^{(j)}(s, x, a)$ is the travel time from x to the next state determined by the choice of driving action a, at traffic flow state s. The expectation in Bellman equation is replaced here by taking n samples $\Delta t^{(j)}(s, x, a)$, $j = 1, \ldots, n$ of the action cost. The k-greedy approach unfolds the Bellman equation up to depth k and then relies on the heuristic. The computation of $\hat{V}$ involves a solution of a relaxed search problem. We look at two of the previous approximate search methods used as heuristics, SNAPSHOT and MEANSPEED. Note that when a relaxed search problem is used as a heuristic, the online method also avoids getting stuck in states from which the goal is unreachable. We define such states x to have a negative-infinity value $V(v, r, x)$ for all v, r. Thus any state from which the goal is reachable (although perhaps not at the estimated cost) will be visited first.

K-beam-greedy strategy One disadvantage of the greedy strategy is its short-sightedness. Depending on the quality of the heuristic approximation $\hat{V}$, the agent is more or less likely to wind up in an unfavorable region of the state space. We propose endowing the greedy strategy with the advantage of far-sightedness of the BEST-OF-k method. At a decision point, we obtain a new state update s_{new} and proceed to generate K state-trajectory samples beginning in s_{new}. Then, if and only if the paths propose different first steps we proceed to evaluate them on M samples as in BEST-OF-K. We call the combined method K-BEAM-GREEDY.

The structure of the algorithm also makes it more amenable to give it the desirable "anytime" property. The main computational hurdle is the prediction of the future traffic state using the dynamic state model $p(s'|s)$. Incrementally sampling trajectories from the dynamic model helps to optimally utilize the time between decision points.

5 Experiments and discussion

In this section, we describe the particulars of the simulation model parameterization and report the obtained results.

Network The simulated traffic network with 61 nodes and 144 links is shown in Table 1. The flow–distribution parameters θ_{ij} were chosen so that the volume distributes evenly

[3] Alternatively, the evaluation could just as well be done with the $K - 1$ remaining trajectories.

TIME-DEPENDENT-SHORTEST-PATH
Input: states $orig, dest$, trajectory $\mathbf{t}$
Output: path from $orig$ to $dest$
$root := make - node(orig)$;
$push(q, root)$;
while $\neg empty(q) \wedge \neg found$ **do**
  $active := top(q)$;
  $children :=$TIME-DEPENDENT-EXPAND$(active, \mathbf{t})$;
  $push(q, children)$;
end

TIME-DEPENDENT-EXPAND
Input: node n, trajectory $\mathbf{t}$
Output: set of children nodes
$x := state(n)$;
foreach c *in* $Succ(x)$ **do**
  $n_c := make - node(c)$;
  $g(n_c) := g(n) + time(x, c, \mathbf{t}(n))$;
end

Algorithm 1: Time-dependent shortest path. The algorithm steps are reminiscent of a standard shortest-path method. The difference is that the node expansion of the search-tree nodes calculates the travel-time for the new segment from the given traffic state trajectory.

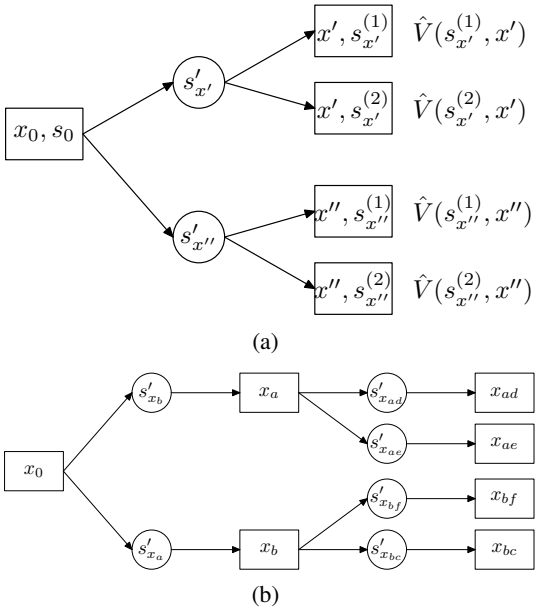


Figure 2: b) The greedy method expands the decision diagram for one or more levels and relies on the heuristic $\hat{V}$ at a preset depth. b) The k-BEAM-GREEDY search pre-selects the k state trajectories. The method plans with each trajectory separately. Therefore each chance node has only a single child and the stochastic search is reduced to a small number of simple graph search instances.

into successor links: $\theta_{ij} = 1/|Succ(i)|$. The network state was initialized in a randomly generated state and allowed to burn-in for a significant time before sampling. We chose sampling time $\delta t = 0.01$ for the traffic state model. The volume-speed relationship is modeled by fundamental diagrams similar to the one shown in Figure 1a. The model is suggested in Chapter 3 of the Highway Capacity Manual (Transportation Research Board 2000), the authoritative compendium of traffic management methods.

Evaluation The start and end states are fixed throughout the evaluation. The experiment consists of 20 independent rounds, each with a different initial traffic state. For each initial traffic state a set of test traffic-state trajectories is generated according to the model. These trajectories are used to evaluate the performance of each routing algorithm under the conditions defined by the testing state trajectory. The results of our experiments are are tabulated in Table 1.

5.1 Fixed-plan methods

All fixed plan methods are compared against the MAXSPEED method that does not take advantage of the traffic information and uses the simple Dijkstra's shortest path algorithm to obtain the route. MAXSPEED fails to take note of frequent congestion patterns occurring on some roads and leads the agent onto congested roads. The SNAPSHOT algorithm freezes the initial traffic situation to build a plan. It is plagued by failures to account for change in the traffic state and in our experiments performed even slightly worse than the MAXSPEED method. MEANSPEED method is better, but its performance, based on average traffic flow behavior, is limited by the variance of actual traffic costs. The BEST-OF-k method, emerges a winner among the fixed strategies because it is able to predict future possible contingencies better. It ran with $k = 10$ planning and $M = 20$ internal evaluation trajectories. The OMNISCIENT method represents the unattainable lower bound on path cost by planning with foreknowledge of the actual testing situation.

Method	Cost $\pm$ stdev	Runtime
MAXSPEED	7.02 ± 0.635	0 ± 0
SNAPSHOT	7.09 ± 1.14	1.6 ± 4.92
MEANSPEED	6.85 ± 0.662	2.35 ± 5.74
BEST-OF-k	6.30 ± 0.646	30.6 ± 11.8
GREEDY + SNAPSHOT	6.47 ± 0.531	0 ± 0
GREEDY + MEANSPD	6.38 ± 0.443	0 ± 0
K-BEAM GREEDY	6.28 ± 0.693	9.7 ± 2.51
OMNISCIENT	6.00 ± 0.331	0.8 ± 3.58

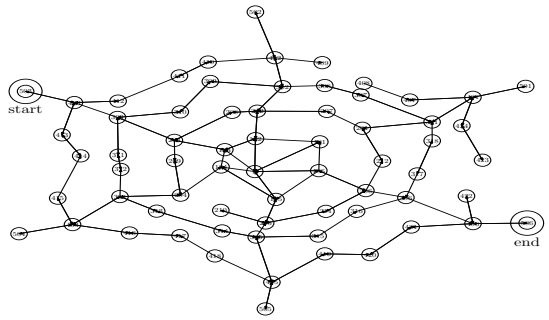


Table 1: Results on the simulated traffic road network, finding path from *start* to *end*. (opposing peripheral positions). The cost (travel time) is in minutes. Running time is in milliseconds. For the fixed strategies, the time is that of the execution of the algorithm; for the information feedback strategies, it is the average time to reach a decision at an intersection. Where time measurements are 0, the method was too fast to measure time accurately on the millisecond scale.

In terms of runtime, the MAXSPEED algorithm was the best, as expected. MAXSPEED has no need to access the current state data. The SNAPSHOT method only needs to retrieve a single vector of speeds, while MEANSPEED is slowed down by trajectory sampling and the mean computations. BEST-OF-k spends over 10 times more computation as it has to both plan 10 paths and evaluate them.

5.2 Search with feedback

Greedy 1-step look-ahead policies The greedy methods are tested with a 1-step look-ahead. As expected, the performance of the greedy look-ahead methods depends on the heuristic $\hat{V}$. The online search equipped with the MEANSPEED search heuristic had better estimates of the true value than the search using the snapshot heuristic. This parallels the standalone performance difference between the two methods embedded into the greedy policy as heuristics.

Both variants achieved negligible average time to make a decision. This is achieved by caching of heuristic evaluation results – if a state is reached in heuristic evaluation that is identical to a state seen in previous searches, the evaluation is terminated and the previous result is returned.

K-beam greedy policy The performance of the K-BEAM-GREEDY is similar to that of the related BEST-OF-k method in cost. A slight improvement is observed due to the availability of additional information during plan execution, allowing for course corrections. However, the method appears to be driven heavily by the underlying cost estimator. In terms of planning time, the method is faster than BEST-OF-K, because it only evaluates the resulting paths if they disagree on the first step.

6 Summary and future work

The general formulation of the shortest path problem in networks with stochastic time-dependent cost as a special semi-Markov decision process is intractable due to large continuous state spaces. We have examined how well several approximation strategies work and found that planning on a limited set of state-trajectories sampled from a predictive model is a viable method of planning in these environments.

Our study leaves some things to be desired, notably an evaluation on real-world network data with tens of thousands of components that is currently being prepared. We could use a multi-modal characterization of link behavior that more precisely captures traffic states - for instance, to add an incident state representing the mode of traffic that occurs when an obstacle in the flow appears. The models of dynamics, while adequate for our purposes here, can be refined ad infinitum. Our solution relies heavily on sampling as the transition model does not readily lend itself to closed form solution. Certainly, special cases that do permit such solutions deserve to be investigated.

Interesting effects might arise if such route guidance systems are massively adopted and followed. Then a route suggested to a large number of drivers may be overwhelming and may change the traffic pattern. While the "negligible effect" assumption in terms of overall flows is still valid, the fact that many vehicles "see" far-ahead along their potential routes is likely to change the overall model of traffic dynamics.

References

Bander, J. L., and White, C. C. 2002. A heuristic search approach for a nonstationary stochastic shortest path problem with terminal cost. *Transportation Science* 36(2):218–230.

Bellman, R. E., and Dreyfus, S. 1962. *Applied Dynamic Programming*. Princeton: Princeton University Press.

Bertsekas, D. P. 1995. *Dynamic programming and optimal control*. Athena Scientific.

Boyan, J. A., and Littman, M. L. 2001. Exact solutions to Time-Dependent MDPs. In *Advances in Neural Information Processing Systems*. MIT Press.

Bradtke, S. J., and Duff, M. O. 1995. Reinforcement learning methods for continuous-time Markov decision problems. In Tesauro, G.; Touretzky, D.; and Leen, T., eds., *Advances in Neural Information Processing Systems*, volume 7, 393–400. The MIT Press.

Bradtke, S. J. 1994. *Incremental Dynamic Programming for Online Adaptive Optimal Control*. Ph.D. Dissertation, University of Massachusetts.

Das, T. K.; Gosavi, A.; Mahadevan, S.; and Marchalleck, N. 1999. Solving semi-markov decision problems using average re-

ward reinforcement learning. *Management Science* 45(4):560–574.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269–271.

Fu, L.; Sun, D.; and Rilett, L. R. 2006. Heuristic shortest path algorithms for transportation applications: state of the art. *Computers and Operations Research* 33(11):3324–3343.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2001. *Elements of Statistical Learning*. Springer.

Howard, R. A. 1963. Semi-Markovian decision processes. In *Proceedings of International Statistical Inst.*

Jewell, W. S. 1963. Markov renewal programming: I. formulations, finite return models II. infinite return models example. *Operations Research* 11:938–971.

Kerner, B. 2003. Dependence of empirical fundamental diagram on spatial-temporal traffic patterns features. *ArXiv Condensed Matter e-prints*.

Kotsialos, A.; Papageorgiou, M.; Diakaki, C.; Pavlis, Y.; and Middelham, F. 2002. Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool metanet. *IEEE Transactions on intelligent transportation systems* 3(4):282–292.

Nagel, K., and Rasmussen, S. 1994. Traffic at the edge of chaos. In Brooks, R. A., and Maes, P., eds., *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, 222. MIT Press.

Pallottino, S., and Scutella, M. G. 2003. A new algorithm for reoptimizing shortest paths when the arc costs change. *Operations Research Letters* 31(2):149–160.

Puterman, M. L. 1994. *Markov decision processes: discrete stochastic dynamic programming*. New York: John Wiley.

Transportation Research Board. 2000. Highway capacity manual. Technical report, Transportation Research Board.