# Route Planning under Uncertainty: The Canadian Traveller Problem

**Evdokia Nikolova** and **David R. Karger** *

## Abstract

The Canadian Traveller problem is a stochastic shortest paths problem in which one learns the cost of an edge only when arriving at one of its endpoints. The goal is to find an optimal policy that minimizes the expected cost of travel. The problem is known to be #P-hard. Since there has been no significant progress on approximation algorithms for several decades, we have chosen to seek out special cases for which exact solutions exist, in the hope of demonstrating techniques that could lead to further progress. Applying a mix of techniques from algorithm analysis and the theory of Markov Decision Processes, we provide efficient exact algorithms for directed acyclic graphs and (undirected) graphs of disjoint paths from source to destination with random two-valued edge costs. We also give worst-case performance analysis and experimental data for two natural heuristics.

## Introduction

The Canadian traveller problem was first defined (Papadimitriou & Yannakakis 1991) to describe a situation commonly encountered by travellers in Canada: once a driver reaches an intersection, he sees whether the incident roads are snowed out or not and consequently decides which road to take. In a graph instance, we are given distributions for the costs of the edges, and upon arriving at a node we see the actual cost values of incident edges. The goal is to find an optimal *policy* for reaching from source $S$ to destination $T$ that minimizes expected cost.

The Canadian Traveller problem is exemplar of several interesting and important issues in optimization. It is a *stochastic optimization* problem, in which some details of the input (the edge lengths) are unknown when the problem is being solved. It is *adaptive*—decisions are made as the algorithm is running, based on new information that arrives during that time. There is a *cost to gather information*—one must travel to a vertex to discover its incident edge costs. It therefore involves the classic problem of *exploration versus exploitation*, in which one must decide whether to exploit the (possibly suboptimal) information acquired so far, or invest further cost in exploration in the hope of acquiring better information.

The Canadian traveller problem and similar formulations find application in transportation, planning, robot navigation and other areas. Its importance is convincingly demonstrated by the diversity of fields of research that have considered it, eg. (Papadimitriou & Yannakakis 1991; Ferguson, Stentz, & Thrun 2004; Blei & Kaelbling 1999; Bar-Noy & Schieber 1991; Polychronopoulos & Tsitsiklis 1996). Despite its importance, very little progress has been made in formal approaches to the problem. Canadian Traveller was shown to be #P-hard (Papadimitriou & Yannakakis 1991). Because of this negative result, the focus falls naturally on approximation algorithms. But to date, none have been developed. Indeed, even simpler questions such as whether there exists a polynomial size *description* of an optimal or approximately optimal policy, regardless of whether it can be constructed, remain unanswered.

**Our results.** In light of the lack of progress on approximation algorithms, we have chosen to explore *exact* solutions on special classes of graphs. Our aim is to begin to understand the structure of good solutions, and the barriers to finding them, as a first step toward more general instances. We explain the connection of our problem to *Markov Decision Processes (MDPs)*. MDPs can be solved in time polynomial in the size of the state space. The problem is that for Canadian Traveller, the obvious state space is exponential in the size of the graph: there is one state for each possible set of observations of values (so far) for any reachable subset of the edges.

We begin with the (well understood) result that if incident edge costs are resampled each time we visit a vertex, the problem becomes a standard MDP solvable

*MIT CSAIL, Cambridge MA 02139, USA, {enikolova, karger}@csail.mit.edu

in polynomial time. Similarly, in the case of a directed acyclic graph (DAG), the problem is easily solved by dynamic programming. In both cases, the state space is small because we need not remember edge costs—in the case of resampling, the edges will be different each time we look, and in the case of DAGs, we will never return to an edge. But as soon as one can return to edges of fixed value, this approach fails.

It became clear that a core question in Canadian traveller is when to turn back and seek a different route. To separate this question from the more general problem of which way to go forward, we considered the special case of a graph made of parallel paths (which are node disjoint except at the source and destination). We give an exact polynomial-time algorithm for the case where the edges take on two distinct values at random. When one of the values is 0 the optimum policy has a simple closed form description; when both values are nonzero we show how to capture the problem in a *small* MDP. Finally, we analyze and provide experimental results for two natural heuristics. A number of the proofs appear in the full version of this paper (Karger & Nikolova 2008).

**Additional related work.** Since each field has used a different term for the same problem such as the *bridge problem* (Blei & Kaelbling 1999) and others, related work is sometimes difficult to recognize. Among the work most closely related to ours after Papadimitriou & Yannakakis (Papadimitriou & Yannakakis 1991), is that of Bar-Noy and Schieber (Bar-Noy & Schieber 1991) who analyze an adversarial version of the problem from a worst-case perspective. The authors also consider a stochastic version, though overall their assumptions alter the nature of the problem significantly and allow for an efficient solution. We assume that once an edge cost is observed, it remains fixed for all time and remark in the next section on the much easier version in which a new value for the edge is resampled from its distribution at each new visit. Blei and Kaelbing (Blei & Kaelbling 1999) consider the same version of the Canadian traveller problem discussed here and offer solutions based on Markov Decision Processes, though they note that their solutions are exponential in the worst case.

A different perspective of adaptive routing problems is offered by the framework of competitive analysis (Kalai & Vempala 2005). Other work on adaptive routing includes (Fan, Kalaba, & J. E. Moore ; Gao & Chabini 2002; Boyan & Mitzenmacher 2001; Hajiaghayi *et al.* 2006; McDonald ; Peshkin & Savova 2002).

## Problem formulation and preliminaries

Consider a graph $G(V, E)$ with a specified source $S$ and destination $T$. The edges in the graph have uncertain costs, coming from known probability distributions. For edge $e$, we shall denote its random variable cost by $X_e$ and its realized value by $x_e$ or simply $cost(e)$.[1] Upon reaching a node, an agent observes the realized values of all edges adjacent to that node. The objective is to find the optimal policy for routing from the source to the destination, which minimizes the expected cost of the route.

### A Markov Decision Process formulation

It is known that the optimal policy of Markov Decision Processes (MDPs) can be found in polynomial time in the size of the MDP (the number of states and actions), for example via linear programming (Puterman 1994). However, most often, the size of the MDP blows up and such exact solutions are typically exponential, unless there is a clever way of defining the MDP which reduces its size.

The Canadian traveller problem can be reduced to a finite belief-state MDP in which a state is a node together with values of observed edges so far. Note however that due to the unfolding history this would lead to exponentially many states even on a simple graph with only two possible values for the edge costs and thus it would give an inefficient solution for the optimal policy.

### Comparison with an easier version: Canadian Traveller with resampling

Consider the Canadian Traveller problem as defined above with the additional assumption that every time we come back to a node, we observe newly resampled values for the incident edges. In this case, there a natural choice for an MDP that solves the problem in polynomial time.

Define an MDP in which is a state is simply the node of current position. An action is the choice of which next node to visit. Then the number of states is equal to the number of vertices $|V|$ and the MDP can be solved in polynomial time: in particular, we can find in polynomial time the expected cost $w(v)$ of the optimal policy for getting from each node $v$ to the destination (prior to having seen the costs of edges incident to $v$). The optimal action once we arrive at a node $v$ is then to choose its next neighbor $v'$ minimizing the cost of edge $(v, v')$ plus the optimal cost $w(v')$ to the destination.

## Optimal policy for DAGs

For the rest of the paper, we consider the standard version of the Canadian Traveller problem, in which once observed, an edge cost remains fixed. In this section we show that the special case of *directed acyclic graphs* (DAGs) can be solved in polynomial time.

In general the Canadian Traveller problem with fixed observations is much harder than the version with resampling: perhaps surprisingly, since in essence we are

[1]We distinguish between the cost of a path, which is the sum of the costs of its edges, and its length, which is the number of edges on the path.

now better informed of the costs of all previously observed edges. From the point of view of MDPs, a corresponding MDP modeling this version requires a much bigger state space since a state needs to reflect all past observations. An exact solution would therefore require exponential time.

In a DAG on the other hand, we can never return to an edge that has been previously visited, so the versions with and without resampling are essentially identical problems. Thus we can solve DAGs in polynomial time with the same MDP as in the case of resampling. We now give a direct faster polynomial-time solution for general DAGs based on dynamic programming.

**Theorem 1.** *The Canadian traveller problem on a directed acyclic graph with $|E|$ edges can be solved exactly in time $O(|E|)$.*

*Proof.* Denote by $w(v)$ the expected cost of following the optimal policy from node $v$ to the destination. Upon arrival at node $v$, one sees the actual costs of all outgoing edges, hence the optimal policy would choose the edge $(v, v')$ minimizing $\{cost(v, v') + E[cost(v', destination)]\}$. The second term is $w(v')$ by definition. Thus, the expected cost at $v$ is the expectation of this minimum:

$$w(v) = \mathbf{E}[\min_{v'}\{X_{vv'} + w(v')\}], \quad (1)$$

where $X_{vv'}$ is the random cost of edge $(v, v')$.

From Eq. (1), the optimal policy costs $w(v)$ can be computed efficiently by traversing the nodes of the graph $v$ in reverse topological order. The calculation of a minimum of several random variables is standard in statistics. In the graph, we can compute the expected minimum as $p_1c_1 + (1 - p_1)[p_2c_2 + (1 - p_2)[p_3c_3 + ...]]$, where $c_1 < c_2 < ...$ are the possible values of the random variables $[X_{vv'} + w(v')]$ (for all $v'$) and $p_1, p_2, ...$ their corresponding probabilities. Thus computing $w(v)$ for a single node from Eq. (1) is done in time linear in the number of outgoing edges of $v$. Consequently, the total running time of the algorithm is linear in the total number of edges $O(|E|)$.

The optimality of this algorithm critically follows from the fact that the graph is directed and acyclic, thus the optimal policy cost $w(v)$ at every node $v$ only depends on the children of $v$ and hence it is computed correctly (proof by induction, starting from the last node). □

## Properties of disjoint-path graphs

In this section, we derive a key monotonicity property of the optimal policy on undirected disjoint-path graphs (consisting of disjoint $ST$-paths), with independent identically distributed (*iid*) edges.

Consider an undirected graph consisting of $k$ node-disjoint paths between the source $S$ and destination $T$. Any algorithm would follow one path up to a point, then possibly return to the source to explore a part of a second path, etc. until it reaches the destination. In the worst case, it may turn at every node, reaching the next unexplored node on another path, then turning back to reach the next unexplored node on a different path, etc. At every step of the algorithm we arrive at a configuration where we know the explored distance $a_i$ and the number of unexplored edges $n_i$ on the $i$-th path for $i = 1, ..., k$, as shown in Figure 1. In this configuration, we denote the expected cost to the destination under the optimal policy by $w(\{a_i, n_i\}_{i=1}^k)$. With a slight abuse of notation, we will use the same notation for a configuration with current position inside a path. Then for a different path $i$, $a_i$ will stand for the sum of distances from the current location back to the source and from the source to the first unexplored edge on path $i$.

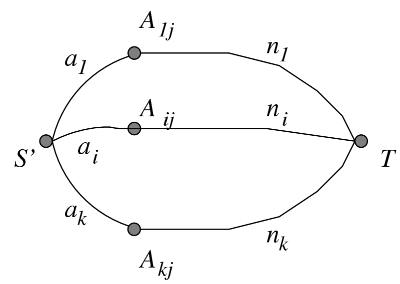


Figure 1: $w(\{a_i, n_i\}_{i=1}^k)$ is the optimal expected cost to get from $S'$ to $T$.

**Lemma 2.** *[Monotonicity] The following properties hold for the optimal algorithm and its expected cost $w(\{a_i, n_i\}_{i=1}^k)$:*

1. *The optimal expected cost is symmetric with respect to the paths, $w(a_i, n_i, a_j, n_j, ...) = w(a_j, n_j, a_i, n_i, ...)$.*
2. *The optimal expected cost is monotone nondecreasing in $a_i$ and $n_i$, for all $i = 1, ..., k$.*

*Proof.* Part (1) follows from the isomorphism of the graphs with the $i$th and $j$th paths exchanged. For part (2), we first show that the optimal expected cost is monotone nondecreasing in $n_i$. By symmetry, it suffices to show this for $n_1$. We need to show $w(a_1, n_1, \{a_j, n_j\}_{j\neq1}) \leq w(a_1, n_1 + n, \{a_j, n_j\}_{j\neq1})$ for $n > 0$. Denote the configurations $\mathcal{C}_{n_1} = (a_1, n_1, \{a_j, n_j\}_{j\neq1})$, $\mathcal{C}_{n_1+n} = (a_1, n_1 + n, \{a_j, n_j\}_{j\neq1})$ and $\tilde{\mathcal{C}}_{n_1+n} = (a_1, n_1+n, \{a_j, n_j\}_{j\neq1})$ where the last $n$ edges on the first path have cost 0.

Suppose the optimal algorithm on $\mathcal{C}_{n_1+n}$ is $\mathcal{A}$. Consider algorithm $\mathcal{A}^*$ on configuration $\tilde{\mathcal{C}}_{n_1+n}$ defined as follows:

$(i)$ Run $\mathcal{A}$ until it reaches the destination $T$ or the node $T'$ on the first path, which immediately precedes the $n$

zero edges;
$(ii)$ If reach $T$ before $T'$, terminate;
$(iii)$ Else, proceed from $T'$ to $T$ along the bottom edges at cost 0.

On every realization of the edge random variables in configuration $\mathcal{C}_{n_1+n}$, Algorithm $\mathcal{A}^*$ incurs the same or smaller cost $\tilde{\mathcal{C}}_{n_1+n}$ than Algorithm $\mathcal{A}$ on $\mathcal{C}_{n_1+n}$. On the other hand, the cost of running $\mathcal{A}^*$ on configuration $\tilde{\mathcal{C}}_{n_1+n}$ is precisely the same as the cost of running $\mathcal{A}$ on configuration $\mathcal{C}_{n_1}$. Therefore,

$$\begin{aligned} w(a_1, n_1, \{a_j, n_j\}_{j\neq 1}) &\leq w_{\mathcal{A}}(a_1, n_1, \{a_j, n_j\}_{j\neq 1}) \\ &= w_{\mathcal{A}^*}(\tilde{\mathcal{C}}_{n_1+n}) \\ &\leq w(a_1, n_1+n, \{a_j, n_j\}_{j\neq 1}). \end{aligned}$$

A similar argument shows that the optimal expected cost is monotone non-decreasing in $a_i$ for all $i = 1, ..., k$, QED. □

We can now prove the following theorem (Karger & Nikolova 2008).

**Theorem 3.** *Suppose $a_1 = \min a_i$ and $n_1 = \min n_i$. Suppose also that the cost of every edge is a non-negative random variable. Then it is optimal to proceed along the first path up to the first unseen edge on the path.*

## Optimal policy for disjoint-path graphs

In this section, we give a polynomial-time algorithm for computing the optimal policy on a disjoint-paths graph with *iid* random two-valued edge costs. To build intuition, it is instructive to consider first the case in which one of the values is zero (without loss of generality the other value is 1). In this case, Theorem 3 and Lemma 2 immediately establish that the optimal strategy explores all paths until it reaches edges of cost 1 on each path, at which point it comes back to the path with fewest unexplored edges and follows it until the end.

**Theorem 4.** *The optimal policy on a disjoint-path graph with random $(0, 1)$-edge costs can be found in polynomial time.*

Furthermore we can compute closed-form expressions for the expected cost of the optimal route (Karger & Nikolova 2008).

Next we compute an optimal policy based on MDPs for the case of positive edge costs. Without loss of generality let the edge cost be 1 with probability $p$ and $K$ otherwise. Defining MDPs in the natural way yields exponentially large state space as explained above; combining it with the special structure of the optimal policy here lets us define a more parsimonious MDP that is efficiently solvable.

A property of the optimal policy here is that once we have crossed a $K$-edge, we will never cross it again. This follows from a distribution-dominance argument as in our Monotonicity Lemma 2. Similarly, once we have chosen to follow a path, the optimal policy would keep following it until the first $K$-edge.

These properties allow for a simple description of the optimal policy: explore one or more of the paths up to the first cost-$K$ edge along them, then pick one and follow it until the end. The policy just has to decide in what order to explore the paths, and how many, before committing to a path. This allows us to capture it as the solution of an MDP with concise representation.

**Two paths** In order to solve the general case, the optimal policy needs a subroutine for comparing two paths.

**Lemma 5.** *The optimal policy on a graph with two node-disjoint paths and positive two-valued edge costs can be found in polynomial time.*

*Proof.* At each instant prior to committing to a path, our knowledge on the graph can be described concisely as the tuple $(a_1, x_1, n_1; a_2, x_2, n_2; i)$, where $a_i$ is the number of cost-1 edges we have observed along path $i$; $x_i = 1$ or $K$ is the last observation on that path; $n_i$ is the number of unobserved edges remaining on the path and $i = 1, 2$ is the index of the current path. Clearly these are polynomially many $O(|E|^4)$ states and in each state there are only two actions: to continue along the current path or turn back to the other path. ($|E|$ is the number of edges in the graph, as usual.) Thus, the optimal policy can be obtained as the solution to that MDP in polynomial time. □

As a corollary from the two paths case, we have an efficient subroutine for comparing two paths with configurations $(a_i, x_i, n_i)$ using the notation above, in terms of which one would yield a lower expected policy cost.

**Arbitrary many paths** We can now derive the optimal policy for the general case of more than two paths.

**Theorem 6.** *The optimal policy on a disjoint-path graph with positive two-valued edge costs can be found in polynomial time.*

## Series-parallel graphs

In this section we show that we may sometimes turn back along a high-cost edge in a series-parallel graph, so the optimal policy for disjoint-path graphs no longer applies here.[2]

Consider the series-parallel graph in Figure 2, in which an agent starting at $S$ has a choice of two directions, to $A$ or $B$. The graph from $A$ to $T$ consists of $n$ paths with $\log n$ edges each, and there is a single edge from $B$ to $T$. Further suppose that each edge is 0 or 1 with probability $\frac{1}{2}$ each.

[2] A series-parallel graph is defined by the following procedure: starting from a single edge, replace an edge by two edges in parallel or two edges in series and repeat as desired.

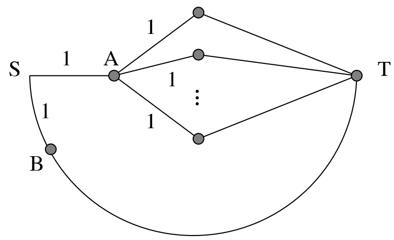


Figure 2: The optimal policy may return along a traversed edge of cost $1$.

**Lemma 7.** *The optimal routing strategy on the given graph first goes to $A$, and upon observing all cost 1 edges at $A$, goes back to $S$, then towards $B$ and $T$.*

## Heuristic algorithms

In the absence of an efficient algorithm for the Canadian Traveller problem on general graphs, we examine two natural heuristics for the optimal routing policy. In the following, we use cost and distance interchangeably. The *minimum expected distance* heuristic replaces unknown edge costs by their expectation and chooses to go to the next node along the shortest path to the destination with respect to these expected edge costs. The *expected minimum distance* heuristic goes to the next node, which minimizes the sum of the corresponding (observed) edge cost and the *expected minimum* distance from the next node to the destination. In other words, for every given realization of edge costs we pick the minimum-cost route; then we calculate the expectation of the minimum-cost route with respect to the distributions of all unknown edge costs in the graph. Note that for different edge cost realizations the minimum route may be different, thus it is meaningful to speak of the expected value of the minimum route, but not of an "expected minimum route" per se. The difference between these two heuristics can be illustrated on the graph in Figure 3. At the source, the first heuristic would replace all $X_i$ by their expectations and take the top single-edge route since its expected cost $\min_i E[X_i] - \epsilon$ is smaller than that of any of the bottom routes. On the other hand, the second heuristic would calculate $E[\min_i X_i]$ as the expected minimum cost of the $n$ parallel edges on the bottom, compare this to the expected minimum cost $\min_i E[X_i] - \epsilon$ on the top and choose to traverse the $0$ edge on the bottom. Then, upon reaching the $n$ parallel edges, it will see their actual costs and pick the edge of minimum cost.

**Minimum expected distance heuristic.** We will show that this heuristic has an exponential gap from the optimal routing algorithm, that is it may result in a route with an exponentially higher cost than that of the optimal route, and should therefore be avoided.

**Lemma 8.** *The minimum expected distance heuristic has exponential gap from the optimal policy.*

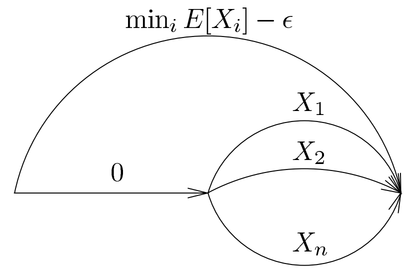


Figure 3: Counterexample showing suboptimality of the minimum expected distance heuristic.

*Proof.* Consider the graph in Figure 3. Any route from the source to the destination consists of the direct link on the top or of the $0$-cost link on the bottom and one of the subsequent $n$ parallel links, whose costs are given by the random variables $X_1, ..., X_n$. Suppose the direct link on the top has cost $\min_i E[X_i] - \epsilon$.

Suppose $X_i$ are random variables, which are $1$ with probability $p > 0$ and $0$ otherwise. If at least one of the $n$ links on the bottom is $0$, the bottom route would have cost $0$, otherwise it would have cost $1$. Thus, $\mathbf{E}[\min X_i] = p^n$. On the other hand, if we follow a partially adaptive algorithm which replaces the unseen edges with their expectation, we would take the top route and the expected cost of the trip would be $\min \mathbf{E}[X_i] - \epsilon = p - \epsilon$. With $\epsilon \to 0$, we see that this gap is exponential:

$$\frac{\min \mathbf{E}[X_i] - \epsilon}{\mathbf{E}[\min X_i]} = \frac{p}{p^n} = \frac{1}{p^{n-1}}.$$

□

**Expected minimum distance heuristic.** We show that this heuristic may yield $(\log n)$-suboptimal routes, although it is significantly better and coincides with the optimal policy on disjoint-path graphs. Note that the expected minimum distance can be approximated arbitrarily well in polynomial time via sampling, and our experimental results demonstrate that even poorer approximations from very few samples can yield an unexpectedly good practical performance. The proofs for the following two lemmas are in the appendix.

**Lemma 9.** *The expected minimum distance rule is optimal on disjoint-path graphs with $\{0, 1\}$ edge costs.*

**Lemma 10.** *The expected minimum distance heuristic has $\Omega(\log |V|)$ gap from the optimal policy.*

**Experimental evaluation.** We implemented the two heuristics and the optimal policy for $n \times n$ directed grid graphs in which all edges are directed either up or to the right, and the goal is to reach from the bottom left to the top right corner of the grid. We chose

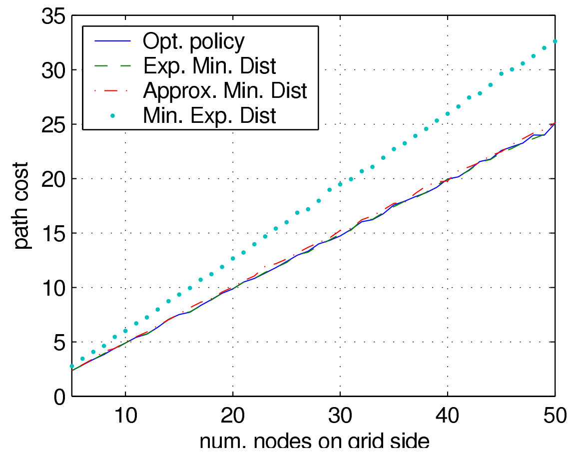


Figure 4: The cost of paths found via the optimal policy and our heuristic algorithms on grid graphs.

these graphs since we could compute the optimal policy efficiently with our algorithm from Theorem 1. This enabled us to compare the heuristics to the optimal solution. These graphs are also fairly realistic, resembling the Manhattan grid of streets. The results for grids of sides $5, 6, \ldots, 50$ are presented in Figure 4. For the purpose of the simulation, we considered *iid* random $(0, 1)$ edge costs with different probabilities. The plots represent the average cost of 1000 actual routes following the optimal policy and the two heuristics. (The plots in the figure have probability $1/2$ of $0$ edge cost, different probabilities yielded similar relative performance of the algorithms). As expected, the minimum expected distance heuristic performed poorly. Despite the negative theoretical worst-case results above, however, the expected minimum distance heuristic had performance comparable to the optimal policy. Even a crude approximation of the expected minimum distances based on only 10 samples of the graph edge costs (represented by the "Approx. Min. Dist." line) yielded surprisingly good routes that were only slightly longer than those of the optimal policy.

## Conclusion

The current status of the general Canadian traveller problem is still widely open with respect to approximability. In light of our experimental results, it would be intriguing to provide a theoretic justification for the excellent performance of the expected minimum distance heuristic. This heuristic may also be a fruitful source of approximation algorithms.

It also remains open to see if the graphs we consider can be extended to a more general class that admits exact polynomial-time solutions, or otherwise to provide hardness results for generalizations of these classes, for example for series-parallel graphs. Given the importance of the problem in practice, it is also key to further understand the effectiveness and limitations of ours and other heuristics, in theory and practice.

## Acknowledgement

We thank Shuchi Chawla, Nick Harvey, David Kempe, Vahab Mirrokni, Christos Papadimitriou, John Tsitsiklis and Mihalis Yannakakis for valuable discussions.

## References

Bar-Noy, A., and Schieber, B. 1991. The canadian traveller problem. In *Proceedings of the second annual ACM-SIAM symposium on Discrete algorithms*, 261–270.

Blei, D., and Kaelbling, L. 1999. Shortest paths in a dynamic uncertain domain. In *IJCAI Workshop on Adaptive Spatial Representations of Dynamic Environments*.

Boyan, J., and Mitzenmacher, M. 2001. Improved results for route planning in stochastic transportation networks. *Symposium of Discrete Algorithms*.

Fan, Y.; Kalaba, R.; and J. E. Moore, I. Arriving on time. *Journal of Optimization Theory and Applications* forthcom.

Ferguson, D.; Stentz, A.; and Thrun, S. 2004. PAO* for planning with hidden state. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA).*

Gao, S., and Chabini, I. 2002. Optimal routing policy problems in stochastic time-dependent networks. In *Proceedings of the IEEE 5th International Conference on Intelligent, Transportation Systems*, 549–559.

Hajiaghayi, M. T.; Kleinberg, R. D.; Leighton, F. T.; and Raecke, H. 2006. New lower bounds for oblivious routing in undirected graphs. In *Proceedings of Symposium on Discrete Algorithms*.

Kalai, A., and Vempala, S. 2005. Efficient algorithms for online optimization. *Journal of Computer and System Sciences* 71:291–307.

Karger, D. R., and Nikolova, E. 2008. Exact algorithms for the canadian traveller problem on paths and trees. Technical Report MIT-CSAIL-TR-2008-004, MIT.

McDonald, A. B. Survey of adaptive shortest-path routing in dynamic packet-switched networks. citeseer.ist.psu.edu/mcdonald97survey.html.

Papadimitriou, C., and Yannakakis, M. 1991. Shortest paths without a map. *Theoretical Computer Science* 84:127–150.

Peshkin, L., and Savova, V. 2002. Reinforcement learning for adaptive routing. In *Proceedings of Intnl. Joint Conf. on Neural Networks, IJCNN*.

Polychronopoulos, G., and Tsitsiklis, J. 1996. Stochastic shortest path problems with recourse. *Networks* 27(2):133–143.

Puterman, M. L. 1994. *Markov Decision Processes*. John Wiley and Sons.