Arriving on Time: Routing in a Stochastic Network

Marco Nie and Xing Wu

Civil and Environmental Engineering
Northwestern University

October 10, 2008

Outline

Motivation

Question

When should you leave UIC if you want to catch a flight at O'Hare Airport at 9:30 am on a Monday? Assume that you drive and that Check-in takes about 30 minutes.

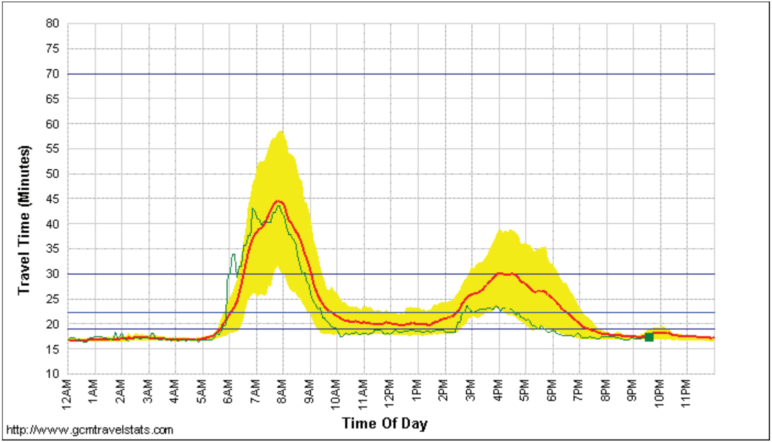
Motivation

Question

When should you leave UIC if you want to catch a flight at O'Hare Airport at 9:30 am on a Monday? Assume that you drive and that Check-in takes about 30 minutes.

- Both Yahoo and Google give the shortest route of about 19 miles. Yahoo estimates the travel time to be 23 minutes, while Google says it will take 26 minutes (but it can be as high as 40 minutes in traffic).
- Do you trust these estimation? If not, how much time you would budget for this trip?

This is a random world!



IL: WB Kennedy from I-290/Circle to O'Hare (17.8 miles)

Yellow area represent normal range: approximately 68% of all travel times will occur within the yellow fill area

Problem statement

Assumptions

- We know the probabilistic distributions of travel times on all roads.
- These distributions are constant across time.
- These distributions are independent to each other.

Problem statement

Assumptions

- We know the probabilistic distributions of travel times on all roads.
- These distributions are constant across time.
- These distributions are independent to each other.

Problem

Find an optimal adaptive or *a priori* path that requires smallest time budget to ensure arriving on-time or earlier for a desired likelihood.

Literature

Expectation-based

- Time-dependent networks: Hall (1986a), Fu (2001), Miller-hooks (2001), Fu & Rilett 1998, Miller-hooks & Mahmassani 2000.
- Correlated distributions: Waller & Ziliaskopoulos (2002), Fan et al. (2005b)
- With recourse: Croucher (1978), Andreatta & Romeo (1988),Polychronopoulos & Tsitsiklis (1996), Waller & Ziliaskopoulos (2002), Provan (2003), Gao & Chabini (2006)

Literature (cont.)

Reliability-based

- Maximizes the probability of realizing a travel time equal to or less than a given threshold: Frank (1969), Mirchandani (1976), Fan et al. (2005a)
- Maximize the probability of being the shortest: Sigal et al. (1980)
- Least possible travel time: Miller-hooks & Mahmassani (1998)
- Maximize expected utility (often leads to bi-criteria problems involving variance): Loui (1983),Eiger et al. (1985), Murthy & Sarkar (1998)
- Minimize the maximum travel time: Yu & Yang (1998), Montemani & Gambardella (2004)

Literature (cont.)

Reliability-based

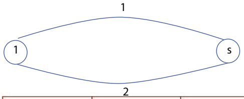
- Maximizes the probability of realizing a travel time equal to or less than a given threshold: Frank (1969), Mirchandani (1976), Fan et al. (2005a)
- Maximize the probability of being the shortest: Sigal et al. (1980)
- Least possible travel time: Miller-hooks & Mahmassani (1998)
- Maximize expected utility (often leads to bi-criteria problems involving variance): Loui (1983),Eiger et al. (1985), Murthy & Sarkar (1998)
- Minimize the maximum travel time: Yu & Yang (1998), Montemani & Gambardella (2004)

Setting

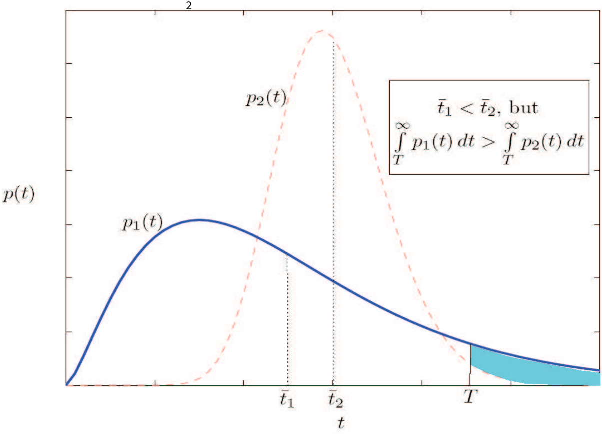
Notation

- Consider A directed network $G(\mathcal{N}, \mathcal{A}, \mathcal{P})$ consisting a set of nodes $\mathcal{N}$ ($|\mathcal{N}| = n$), a set of links $\mathcal{A}$ ($|\mathcal{A}| = m$), a probability distribution $\mathcal{P}$ describing the statistics of the link traversal times (or costs).
- The traversal times of link ij (denoted as c_{ij}) is an independent random variable, following a given distribution $p_{ij}(\cdot)$.
- Cost on path k^{rs} (which connects node r and the destination s) is denoted as π_k^{rs} and all paths that connect r and s forms a set of K^{rs}.
- The destination of routing is denoted as s.

A simple example



Route 2 gives a higher probability of arriving node s within T, but a worse expectation.
To achieve the same reliability, a larger T should be budgeted if you decide to travel on Route 1.



Adaptive path problem (Fan 2005a)

Definitions

Let

- $u_i(b)$ be the maximum probability of arriving node s on time or earlier, starting from node i with a time budget b.
- $k_i(b)$ be the next node to visit at node i in order to achieve $u_i(b)$.

Adaptive path problem (Fan 2005a)

Definitions

Let

- $u_i(b)$ be the maximum probability of arriving node s on time or earlier, starting from node i with a time budget b.
- $k_i(b)$ be the next node to visit at node i in order to achieve $u_i(b)$.

Facts

- If a traveler at node i with a time budget b chooses to visit node j next, the probability that the traveler spends a time in the interval $[w, w + dw]$ is $p_{ij}(w)dw$.
- Once s/he arrives node j, the time budget for the remaining journey becomes $b - w$.

A dynamic programming formulation

Bellman's principle of optimality

No matter which node j the traveler selects to go, s/he must follow the optimal routing strategy traveling from that node j to the destination within time $b-w$.

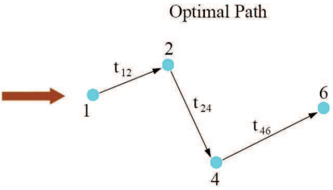
$$u_i(b) = \max_j \int_0^b p_{ij}(w) u_j(b-w) dw, \forall i \neq s, b \geq 0 \quad \text{(1a)}$$

$$u_s(b) = 1, \forall b \geq 0 \quad \text{(1b)}$$

$u_i(b)$ and $k_i(b)$ can be solved from the above system of integral equations for any i and any b from 0 up to an upper bound T.

An illustration of adaptive routing

Node \ Time	T	.....	$T - t_{12}$	.....	$T - t_{12} - t_{24}$	.....
1	2					
2			4			
3						
4					6	
⋮						
N						



The figure shows how the optimal routing policy is used in real-time adaptive routing.

Solution procedures

Successive approximation (Fan, 2005a)

Starting from an initial guess, iteratively update $u_i(b)$ based on the equation.

- Step 0: Initialization. Set $k = 0, u_i^k(b) = 0, i \neq s, b \geq 0$, and $u_s^k(b) = 1, b \geq 0$.
- Step 1: Set $k = k + 1$, $u_i^k(b) = \max_{j \neq i} \int_0^b p_{ij}(w) u_j^{k-1}(t - w) dw, \forall i \in N, i \neq s, b \geq 0.$
- Step 2: Convergence test. If $\max(|u_i^k(b) - u_i^{k-1}(b)|) < \epsilon$, stop; otherwise, go to Step 1.

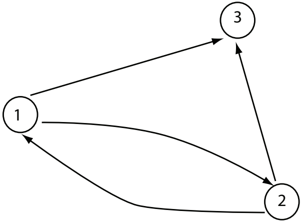
Unlike traditional shortest path problem, however, the number of iterations required is not bounded by the number of nodes (because an optimal routing policy may contain loops).

Solution procedures (cont.)

A pseudo-polynomial algorithm (Nie and Fan 2006)

- Works for the discrete version of the problem.
- Updates $u_i(b)$ following an increasing order of time budgets.
- Runs in a pseudo-polynomial time of $O(mL^2)$.

A numerical example



Link	Traversal time t (minutes) and the associated proba	
1-2	t= 1, $p_{ij}(1) = 0.5$	t=6, $p_{ij}(6) = 0.5$
2-1	t=2, $p_{ij}(2) = 0.5$	t=4, $p_{ij}(4) = 0.5$
2-3	t=4, $p_{ij}(4) = 0.1$	t=6, $p_{ij}(6) = 0.9$
1-3	t=2, $p_{ij}(2)$= 0.4	t=12, $p_{ij}(12) = 0.6$

Node 2			Node 1		
Time budget	Optimal probability	Optimal policy	Time budget	Optimal probability	Optimal Policy
1	0.0	N/A	1	0.0	N/A
2	0.0	N/A	2	0.4	3
3	0.0	N/A	3	0.4	3
4	0.2	1	4	0.4	3
5	0.2	1	5	0.4	3
6	1.0	3	6	0.4	3
7	1.0	3	7	0.5	2
8	1.0	3	8	0.5	2
9	1.0	3	9	0.5	2
10	1.0	3	10	0.6	2

Determine a priori optimal paths

Motivation

- Motorists often begin a trip bearing a planned route in mind. Rerouting takes place only if
  - the travel time on the planned one exceeds a certain threshold of tolerance
  - they know how to find a better one
- Motorists often need to make a routing decision before they finish traversing a link.
- Adaptive routing may suggest cycling which is counterintuitive and may not always be feasible

Define optimality

Definition (b-reliable path)

A path k_{rs} is said b-reliable in K^{rs} if and only if $u_k^{rs}(b) \geq u_l^{rs}(b), \forall l$, where $u_k^{rs} = P(\pi_k^{rs} \leq b)$ denotes the cumulative distribution function (CDF) of π_k^{rs}.

Problem statement

A b-reliable path is the path that is most reliable with respect to b. Our problem is to find such reliable paths.

Define optimality

Definition (b-reliable path)

A path k_{rs} is said b-reliable in K^{rs} if and only if $u_k^{rs}(b) \geq u_l^{rs}(b), \forall l$, where $u_k^{rs} = P(\pi_k^{rs} \leq b)$ denotes the cumulative distribution function (CDF) of π_k^{rs}.

Problem statement

A b-reliable path is the path that is most reliable with respect to b. Our problem is to find such reliable paths.

Unfortunately, dynamic programming is not directly applicable because

Theorem

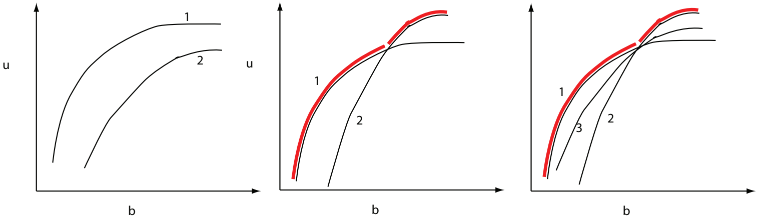
Subpaths of a b-reliable path may not be b-reliable.

Define optimality alternatively

Definition (locally-reliable path)

A path $k^{rs} \in K^{rs}$ is said locally-reliable if and only if $\exists$ no path $l^{rs} \in K^{rs}$ such that 1) $u_l^{rs}(b) \geq u_k^{rs}(b), \forall b$, and 2)$\exists$ at least one open interval $\Lambda \subset [0, \infty]$ with nonzero Lebesgue measure such that $u_k^{rs}(b) < u_l^{rs}(b), \forall l \neq k, b \in \Lambda$.

Obviously, the local optimality is defined in the sense of pareto-optimality or non-dominance relationship.



Path 1 is locally reliable
Path 2 is not. It is domiated by 1
Path 1 forms the pareto frontier

Both Path 1 and 2 are locally reliable
They together form the pareto frontier

All three paths are locally reliable
Path 3 does not contribute to the frontier,
but it is not dominated by either 1 or 2.

Two results

Theorem

Subpaths of any locally-reliable path must be locally-reliable.

Two results

Theorem

Subpaths of any locally-reliable path must be locally-reliable.

So we can still search locally reliable path(s) using dynamic programming. The difference is that we are likely to be left with a set of locally reliable paths, and dealing with such a set can be troublesome.

Two results

Theorem

Subpaths of any locally-reliable path must be locally-reliable.

So we can still search locally reliable path(s) using dynamic programming. The difference is that we are likely to be left with a set of locally reliable paths, and dealing with such a set can be troublesome.

Theorem

A locally-reliable path is acyclic.

Two results

Theorem

Subpaths of any locally-reliable path must be locally-reliable.

So we can still search locally reliable path(s) using dynamic programming. The difference is that we are likely to be left with a set of locally reliable paths, and dealing with such a set can be troublesome.

Theorem

A locally-reliable path is acyclic.

So we can ignore paths with cycles - this may be used to improve computational efficiency.

Solution procedure

Label-correcting

- Step 0: Initialization. Add a path starting and ending at the destination s into candidate list Q.
- Step 1: If Q is not empty, take a path k^{js} from Q, go to step 2; otherwise terminate.
- Step 2: For each path $k^{is} = ij \diamond k^{js}$, if it is locally reliable, add it into Q, and remove all existing paths dominated by this k^{is}. Go back to Step 1.

Theorem (Finite convergence)

The above procedure terminates after a finite number of steps and yields a set of locally reliable paths for each node i.

Complexity

Bad news

Unfortunately, the algorithm is non-polynomial, because the number of locally reliable paths may grow exponentially with the network size. The algorithm runs in order of $O(mn^{2n-1}L + mn^nL^2)$.

We believe that in practice the performance of the algorithm is way better than this worse-case scenario, for reasons.

Complexity

Bad news

Unfortunately, the algorithm is non-polynomial, because the number of locally reliable paths may grow exponentially with the network size. The algorithm runs in order of $O(mn^{2n-1}L + mn^nL^2)$.

We believe that in practice the performance of the algorithm is way better than this worse-case scenario, for reasons.

Good news

- $|K^{is}|$ is much smaller than n^{n-1} for sparse networks commonly seen in transportation applications.
- The expected number of locally reliable paths is bounded roughly by $\log(|K^{is}|)$ if the the number of discrete time points L is 2.

Complexity (cont.)

What if $L > 2$?

Get a theoretical bound is more difficult. However, through experiments we conjecture

- The number of locally reliable paths increases exponentially with L in general, and
- Due to the monotonicity of CDF, it seems to be bounded by $L \log(|K^{is}|)$.

If the second conjecture is correct, we can push the complexity to $O(mn^2L^3(\log(n))^2)$. This is a pseudo-polynomial bound!

Other computational issues

Extreme-dominance approximation

- Ignore locally-reliable paths that do not contribute to the frontier
- The complexity of the solution procedure is now in the order of $O(mnL + mL^3)$ ($\simeq O(mL^3)$). Compared to the adaptive problem, this complexity is only higher by a factor of L.
- This approximation does not always yield correct Pareto-frontiers (it is worthy looking at how it works in practice)
- Bellman's principle of optimality does not work for the set defined by extreme-dominance relationship.

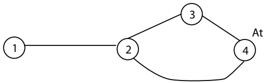
Other computational issues (cont.)

Convolution integral

The calculation of $\int_0^b p_{ij}(w)u_k^{js}(b-w)dw$ is the single most time-consuming component in the solution procedure.

- It has to be evaluated for each path and requires $O(L^2)$ steps to complete.
- For the continuous case, the Laplace transformation may be used to simplify the evaluation (Fan 2005a). But the procedure has to be limited to a few discrete points and may not be stable.
- Another alternative is using the inverse function of CDF (Wu and Nie, 2008a).

Example

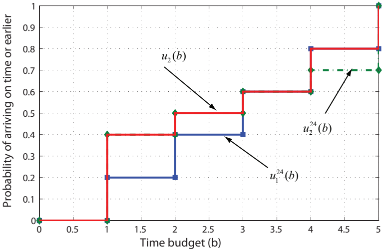


At node 2: path 1 is 2 - 4
path 2 is 2-3-4

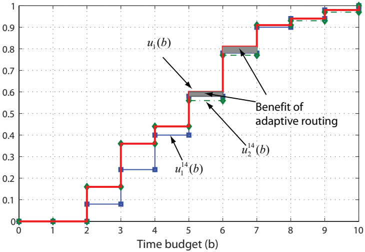
At node 1: path 1 is 1-2-4
path 2 is 1-2-3-4

Time \ Link	1-2	2-4	2-3	3-4
0	0	0	0	1.0
1	0.4	0.2	0.4	0
2	0.4	0.2	0.1	0
3	0	0.2	0.1	0
4	0.1	0.2	0.1	0
5	0.1	0.2	0.3	0

(a)

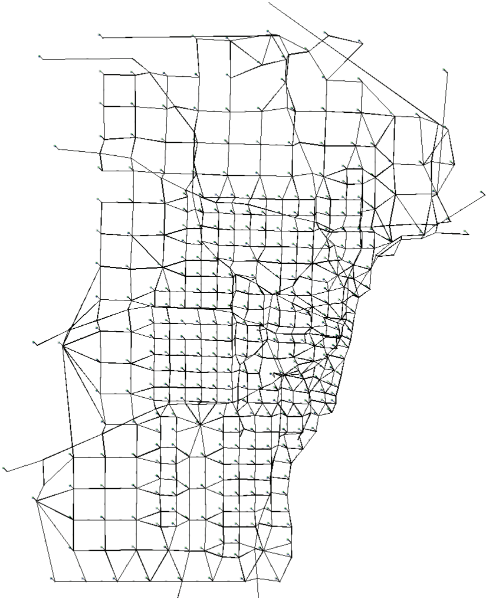


(b)



(c)

A sketch network of Chicago



- The network has 933 nodes and 2950 links. L is set 100.
- Link travel times follow Gamma distribution (parameters are randomly selected within a range.)

Experiment

Compare

- The label-correcting algorithm for the *a priori* path problem and
- Its approximation based on extreme dominance.

Setting

- Run 30 times for different destinations and take average.
- Test three different variance levels (low, medium and high)

Results

Variance	Algorithm	CPU (sec)	Max paths	Ave. paths	Max gap in probability across all nodes		
					30 percentile	60 percentile	90 percentile
Low	LC APPROX	7.4 2.8	42 22	10.3 7.1	6.3E-08	1.1E-07	1.6E-06
Medium	LC APPROX	15.0 3.8	59 26	14.8 8.9	8.7E-08	4.50E-06	9.6E-04
High	LC APPROX	17.9 4.2	69 28	17.0 10.0	2.2E-05	2.4E-04	1.2E-03

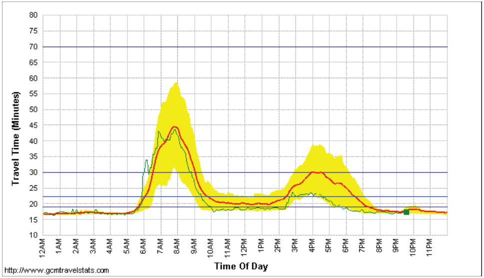
Some findings

- The number of locally reliable paths is reasonably small
- More paths are used for higher variance
- Extreme-dominance offers significant computational benefit, without sacrificing too much on correctness.

Time-dependent case

Motivation

- Link traversal times are not only random, but their distributions also vary over time.
- In transportation, rush hour traffic not only increases the mean travel time, but also leads to larger variance which renders the journey time more unpredictable



A discrete setting

Definitions

- Let time be considered in a discrete manner, i.e., $t = 0, \varphi, 2\varphi, ..., T\varphi$
- Random link traversal time for a vehicle entering into link ij at time t is denoted by c_{ij}^t
- The probability mass function of c_{ij}^t, denoted as $P_{ij}^t(\cdot)$, gives the probability of spending $e\varphi, e = 0, 1, 2, ..., T$ time on link ij when entering at t
- c_{ij}^t can only be a multiplier of φ and $P_{ij}^t(b) = 0, \forall b \notin [0, T]$.

Time-dependent formulation of adaptive path problem

Let $u_i(t,b)$ be the maximum probability of arriving the destination s when starting from node i at t with a budget b

$$u_i(t,b) = \max_j \sum_{h=0}^{b} P_{ij}^t(h) u_j(t+h, b-h), \forall i \neq s, t, b = 0, \varphi, ..., T\varphi \quad (2a)$$

$$u_s(t,b) = 1, \forall b, t \geq 0 \quad (2b)$$

Property of the time-dependent problem

The formulation does NOT increase the dimension of the state space if

- The desirable arrival time is aligned with the end of the departure time horizon
- Both the departure time and the time budget spaces are discretized using the same resolution

Property of the time-dependent problem (cont.)

- Any departure time interval t will uniquely identify a time budget b.
- Time-varying effects are taken into account only through the probability mass function $P_{ij}^{t}(\cdot)$.
- With a slight modification, the time-dependent problem can be solved using algorithms for the static case.
- However, the problem should be solved once corresponding to each desired arrival time

Other extensions

Incorporate correlations

- Assume correlations between adjacent links? (Nie and Wu 2008b)
- Assume a joint discrete distribution is given? (We are working on it.)

Evaluate the model with real traffic data

Currently we are working with Professor Peter Nelson's group to produce real travel time distributions of Chicago area from GCM data.

Thank you.