AAE 706: Applied Risk Analysis

Route Choice with Risk

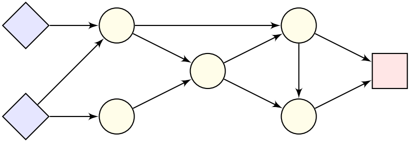
Thomas F. Rutherford

Department of Agricultural and Applied Economics
University of Wisconsin, Madison

12 April, 2023

Minimum-cost flow problems

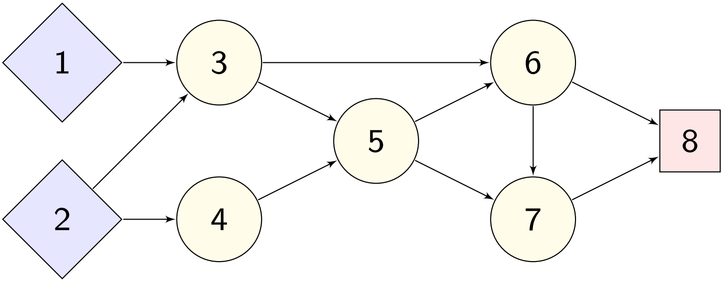
- Many optimization problems can be interpreted as network flow problems on a directed graph.
- Nodes can be sources, relays, or sinks.



- Decision variables: flow on each arc.
- Arcs have flow costs and may have capacity constraints.
- Relays must conserve flow.
- Sources/sinks may have supply/demand limits.

What is the minimum cost feasible flow?

Minimum cost flow problems

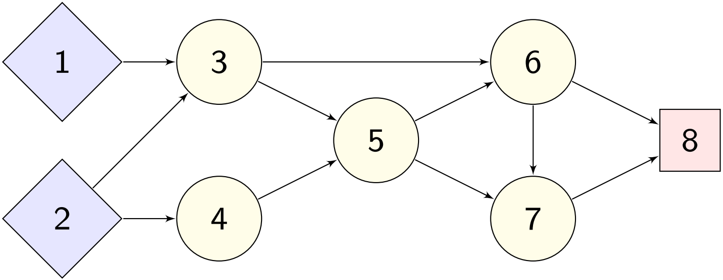


- The set of nodes: $\mathcal{N} = \{1, \dots, 8\}$
- The set of directed arcs: $\mathcal{A} = \{(1,3), (2,3), (2,4), \dots, \}$
- Each node is a source ($\mathcal{S}$), a relay ($\mathcal{R}$), or a sink ($\mathcal{K}$): $\mathcal{S} = \{1,2\}, \mathcal{R} = \{3,4,5,6,7\}, \mathcal{K} = \{8\}$.
- Decision variables: x_{ij} is the flow on arc $(i,j) \in \mathcal{A}$
- Flow cost: c_{ij} is cost per unit of flow on arc $(i,j) \in \mathcal{A}$

GAMS Syntax for Network

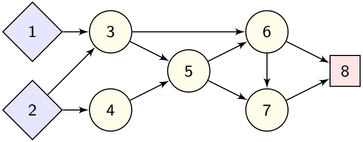
```
set       i       Nodes /1*8/,
          s(i)    Source nodes /1*2/,
          r(i)    Relay nodes /3*7/,
          k(i)    Sink nodes /8/;

alias (i,j,k)

set       a(i,j)  Arcs;

parameter         c(i,j)  Unit cost for flow on edge i-j;

variables         X(i,j)  Flow on edge i-j
                  COST    Total cost of assigned flows;
```

Minimum cost flow problems



- Flow capacity constraints: $p_{ij} \leq x_{ij} \leq q_{ij} \quad \forall ij \in \mathcal{A}$
- Supply constraint: $\sum_{ij \in \mathcal{A}} x_{ij} = s_i \quad \forall i \in \mathcal{S}$
- Demand constraint: $\sum_{ij \in \mathcal{A}} x_{ij} = d_j \quad \forall i \in \mathcal{K}$
- Flow conservation: $\sum_{ik \in \mathcal{A}} x_{ik} = \sum_{kj \in \mathcal{A}} x_{kj} \quad \forall k \in \mathcal{R}$
- Total cost: $\sum_{ij \in \mathcal{A}} c_{ij} x_{ij}$

GAMS Syntax for Network Constraints



- Flow capacity constraints: $p_{ij} \leq x_{ij} \leq q_{ij} \quad \forall ij \in \mathcal{A}$

```
X.LO(a) = p(a);  X.UP(a) = q(a);
```

- Feasible supply constraint: $\sum_{ij \in \mathcal{A}} x_{ij} = s_i \quad \forall i \in \mathcal{S}$

```
fsupply(s(i))..  sum(a(i,j), X(i,j)) =E= supply(i);
```

GAMS Syntax for Network Constraints (cont.)

- Feasible demand constraint: $\sum_{ij \in \mathcal{A}} x_{ij} = d_j \quad \forall i \in \mathcal{K}$

```
fdemand(k(j))..  sum(a(i,j), X(i,j)) =E= demand(j);
```

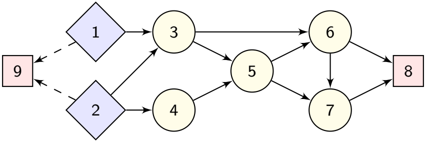
- Flow conservation: $\sum_{ik \in \mathcal{A}} x_{ik} = \sum_{kj \in \mathcal{A}} x_{kj} \quad \forall k \in \mathcal{R}$

```
conservation(r(k))..  sum(a(i,k), X(i,k)) =G= sum(a(k,j),X(k,j));
```

- Total cost: $\sum_{ij \in \mathcal{A}} c_{ij} x_{ij}$

```
totalcost..  cost =e= sum(a, c(a)*X(a))
```

Minimum-cost flow problems



Feasibility of the model requires that aggregate supply equals aggregate demand, i.e. $\sum_i s_i = \sum_j d_j$.

Balancing can be achieved through add a dummy sink node with zero-cost unconstrained links from all sources directly to the dummy.

The new dummy node has no demand constraint.

Supply and demand constraints become equalities:

- Supply constraint: $\sum_j x_{ij} = s_i \quad \forall i \in \mathcal{S}$
- Demand constraint: $\sum_i x_{ij} = d_j \quad \forall j \in \mathcal{K}$

GAMS Code – Illustrative Model

```
$title  Canonical Network Model

set     i       Nodes /1*9/,
        s(i)    Source nodes /1*2/,
        r(i)    Relay nodes /3*7/,
        k(i)    Sink nodes /8/;

alias (i,j,k)

set     a(i,j)  Arcs/
                1.3, 2.3, 2.4, 3.6, 3.5, 4.5,
                5.6, 5.7, 6.7, 6.8, 7.8, 1.9, 2.9 /;

parameter       c(i,j)  Unit cost for flow on edge i-j,
                p(i,j)  Lower bound on flow,
                q(i,j)  Upper bound on flow,
                supply(i)    Supply
                demand(j)    Demand;

*       Generate some random data:

p(a) = 0;
q(a) = +inf;
c(a) = uniform(0,1);
c(i,"9") = 0;
```

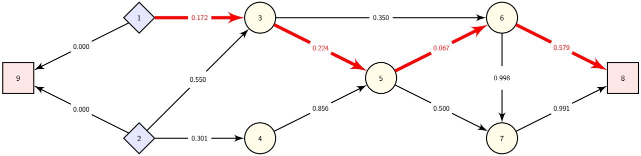
GAMS Code – Illustrative Model

```
variables       X(i,j)  Flow on edge i-j
                COST    Total cost of assigned flows;

equations       supply, demand, conservation, totalcost;

supply(s(i))..  sum(a(i,j), X(i,j)) =L= supply(i);

demand(k(j))..  sum(a(i,j), X(i,j)) =G= demand(j);

conservation(r(k))..  sum(a(i,k), X(i,k)) =G= sum(a(k,j),X(k,j));

totalcost..     COST =e= sum(a, c(a)*X(a));

model network /supply, demand, conservation, totalcost/;

X.LO(a) = 0;  X.UP(a) = +inf;

supply(i) = 1$sameas(i,"1");  demand(j) = 1$sameas(j,"8");

solve network using lp minimizing cost;

parameter       route   Optimal routings;
route(a,"1") = X.L(a);

*        Assign unit supply at node 2, unit demand at node 8:

supply(i) = 1$sameas(i,"2"); demand(j) = 1$sameas(j,"8");
solve network using lp minimizing cost;
route(a,"2") = X.L(a);

option route:0:2:1;  display route;
```

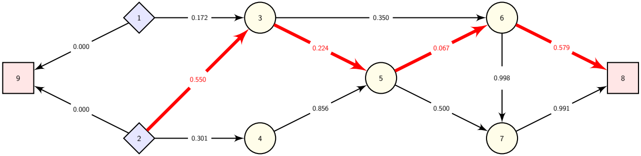
Shortest Paths

```
----     56 PARAMETER route  Optimal routings

                1           2

1.3             1
2.3                         1
3.5             1           1
5.6             1           1
6.8             1           1
```

- Deliver one unit from node 1 to node 8

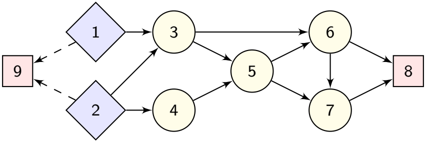


- Deliver one unit from node 2 to node 8



Note: numbers indicate travel cost on each arc in the network.

Minimum cost flow problems

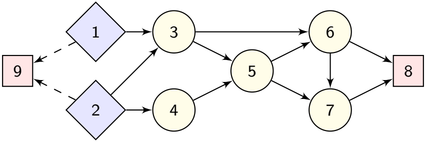


The scalar form of the model reveals the node-arc incidence matrix which connects arc flows (a_{ij}) to supply, demand and conservation constraints ($Ax = b$):

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} x_{13} \\ x_{19} \\ x_{23} \\ x_{24} \\ x_{29} \\ x_{35} \\ x_{36} \\ x_{45} \\ x_{56} \\ x_{57} \\ x_{67} \\ x_{68} \\ x_{78} \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -d_8 \end{bmatrix}$$

Note that no balance condition is included for node 9, as this node is neither a source, a sink nor a relay.

Minimum cost flow problems



The entire model (compact form):

$$
\begin{aligned}
\min_{x \in \mathbb{R}^{|\mathcal{A}|}} \quad & c^T x \\
\text{subject to:} \quad & Ax = b \\
& p \leq x \leq q
\end{aligned}
$$

Note: The incidence matrix A is a property of the graph. It does not depend on which nodes are sources/sinks/relays.

Minimum-cost flow problems

Many problem types are actually min-cost flow models:

- transportation problems
- assignment problems
- transshipment problems
- shortest path problems
- max-flow problems

Shortest path problems

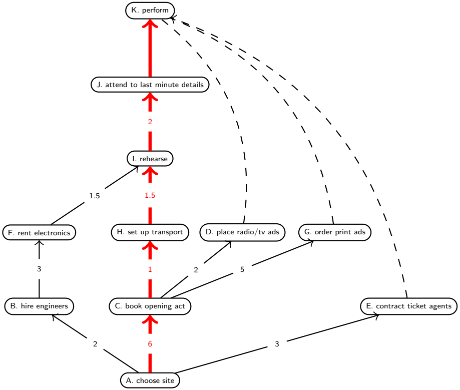
We are given a directed graph and edge lengths. The goal is to find the path with shortest length between two given nodes. This is a special case of a transport

- Again, a transportation problem!
- Edge cost = length of path.
- The source has supply = 1
- The sink has demand = 1

Concert Planning

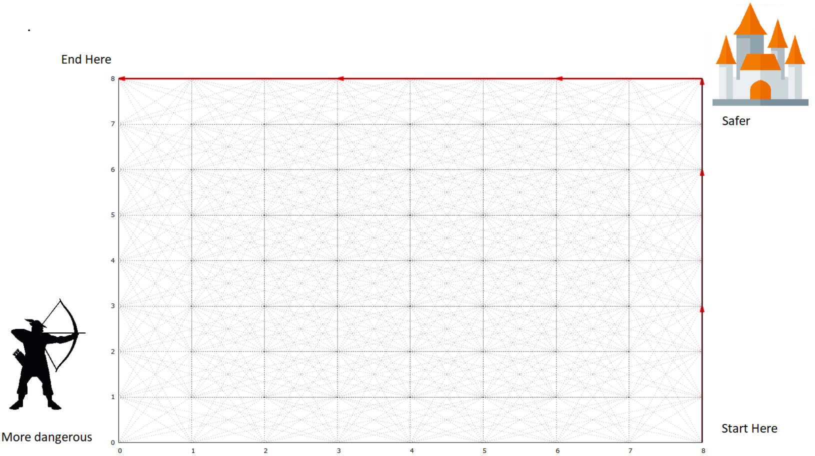
The concert planning example is a *longest path problem*!

- Add source and sink nodes
- Move times out of nodes and onto preceding edges
- Solve longest path problem



The Most Reliable Route: Risk

.



Route Reliability

We have to make a delivery through a road network, starting at node s and ending at node t. We want to devise a route which is both safe and fast. Let p_{ij} be the probability that arc (i, j) can be safely traversed. Assume the probabilities of interuption are independent. The probability that a path P is safely completed is then:

$$\pi_P = \prod_{(ij) \in P} p_{ij}$$

What is the most reliable path from s to t? We can formulate this problem as finding the route which maximizes the probability of reaching t.

A Couple of Optimization Tricks

- Maximizing x is equvalent to minimizing $1/x$:

$$\max \prod_{(i,j)\in P} p_{ij}$$

s.t.

$$P \in \mathcal{P}(s,t)$$

$\Rightarrow$

$$\min \prod_{(i,j)\in P} 1/p_{ij}$$

s.t.

$$P \in \mathcal{P}(s,t)$$

- Because log is a *monotonic* function, optimizing x is equivalent to optimizing $\log(x)$.

$$\begin{aligned} \min_P \log\left(\prod_{(i,j)\in P} 1/p_{ij}\right) &= \min_P \sum_{(i,j)\in P} \log(1/p_{ij}) \\ &= \min_P \sum_{(i,j)\in P} -\log(p_{ij}) \end{aligned}$$

subject to:

$$P \in \mathcal{P}(s,t)$$

Trading off Speed and Reliability

Of course, we are interested in both speed and reliability, so we can form an objective function of the form:

$$\min \sum_{(i,j) \in P} \lambda \tau_{ij} + (1 - \lambda) c_{ij}$$

subject to

$$P \in \mathcal{P}(s, t)$$

where λ is a parameter $(0 \leq \lambda \leq 1)$ representing the relative importance of speed over safety, τ_{ij} is the time cost of arc (i, j) and $c_{ij} = -log(1/p_{ij})$ is the log of the probability of completing the (i, j) arc safely.

Maximum weighted flow formulation

$$\min \sum_{a \in \mathcal{S}} \left(\lambda \tau_a + (1 - \lambda) c_a \right) X_a$$

subject to

$$\sum_{(i,k) \in \mathcal{S}} X_{ik} + 1|_{k=s} = \sum_{(k,j) \in \mathcal{S}} X_{kj} + 1|_{k=t} \quad \forall k \in \mathcal{N}$$

$$X_a \geq 0$$

Aside: Exceptions in GAMS

"\$" is the *such that* operator in GAMS, corresponding to the "|" or "∋" symbols in conventional mathematical notation.

For example, consider the conditional summation:

$$y = \sum_{i \in \{j : \delta_j \neq 0\}} x_i .$$

This is written in GAMS either (verbosely) as

```
y = sum(i$(delta(i)<>0), x(i));
```

or (succintly) as:

```
y = sum(i$delta(i), x(i));
```

Summation over a Subset in GAMS

- If we want to sum over elements in a set $\mathcal{J}$, i.e.:

$$y = \sum_{i \in \mathcal{J}} x_i .$$

this can be written in GAMS with an exception operator as:

```
y = sum(i$J(i), x(i));
```

or we can simply include set $\mathcal{J}$ in the sum:

```
y = sum(J(i), x(i));
```

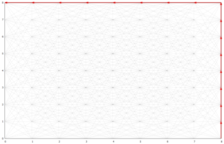
- **N.B.** Arguments to exception condition in a GAMS model *may not* include decision variables.

Maximum Weight Flow Model in GAMS

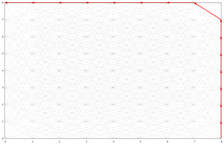
```
variable        OBJ             Objective function;

nonnegative
variables       X               Route choice;

equations       conservation, objdef;

objdef..
        OBJ =e= sum(a, X(a) * dist(a)*(lamda - (1-lamda)*log(p(a))));

conservation(k)..

        sum(a(i,k), X(a)) + 1$start(k) =e= sum(a(k,j),X(a)) + 1$end(k);

model routechoice /conservation, objdef/;
```

Optimal Route Choice

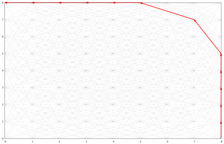
$\lambda = 0.0$



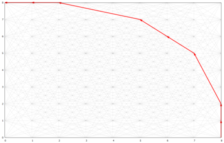
$\lambda = 0.1$



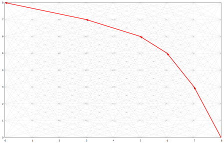
$\lambda = 0.2$



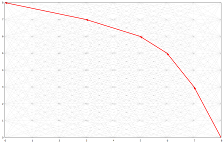
$\lambda = 0.3$



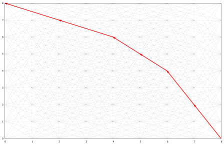
$\lambda = 0.4$



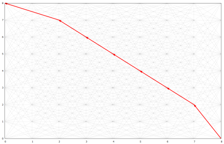
$\lambda = 0.5$



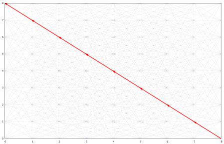
$\lambda = 0.6$



$\lambda = 0.7$



$\lambda = 0.8$



The Architecture of Complexity: Tuples

```
$title  Optimal Route Trades off Travel Time and Reliability

*       An environment variable defines the number of nodes
*       on each side of the square grid:

$if not set n $set n 8

set     r       Rows and columns in the grid /0*%n%/;

*       ir:     Row index for node i
*       ic:     Column index for node i

*       jr:     Row index for node j
*       jc:     Column index for node j

alias (r,ir,jr, c,ic,jc);

*       Short-hand for nodes in the network:

set     i(ir,ic)        Nodes in the network;

*       The grid is dense (but need not be):

i(ir,ic) = yes;
```

The Architecture of Complexity: Tuples

```
*       Define a sequence order for nodes to be used if needed:

parameter       id(ir,ic)       Node identifier (integer),
                nid             ID counter /0/;
loop(i,
        nid = nid + 1;
        id(i) = nid + 1;);

set     a(ir,ic,jr,jc)  Arcs in the network;

parameter       dist    Distance between two nodes;

dist(a(ir,ic,jr,jc)) = sqrt( sqr(ir.val-jr.val)+sqr(ic.val-jc.val));

parameter       pn      "Probability of safe passage is highest at %n%,%n%";

pn(ir,ic) = sqrt(sqr(ir.val)+sqr(ic.val))/sqrt(2*%n%*%n%);

set     start(ir,ic)    Starting point /%n%.0/

        end(ir,ic)      Ending point   /0.%n%/;
```

Maximum Weight Flow Model in GAMS

```
variable        OBJ               Objective function;

nonnegative
variables       X                 Route choice;

equations       conservation, objdef;

objdef..
        OBJ =e= sum(a, X(a) * dist(a)*(lamda - (1-lamda)*log(p(a))));

conservation(k)..

        sum(a(i,k), X(a)) + 1$start(k) =e= sum(a(k,j),X(a)) + 1$end(k);

model routechoice /conservation, objdef/;
```

Loop over λ, Solve and Report

```
set     lamdaval        Set of values of lamda (x10) /0*10/;

set             rc(lamdaval,%i%,%j%)        Chosen routes;

loop(lamdaval,

        lamda = lamdaval.val/10;

        solve routechoice using lp minimizing obj;

        rc(lamdaval,a(i,j))$round(X.L(a),4) = yes;
);
```

Put Statements Generate a GNUPLOT Script

```
file kplt /'figure.plt'/; put kplt; kplt.lw=0;

loop(lamdaval,

...
        loop(rc(lamdaval,ir,ic,jr,jc),
          put 'set arrow from ',ir.tl,',',ic.tl,' to ',jr.tl,',',jc.tl,
                ' head ls 1 lw 3 lc rgb "red"'/;)

        put "set output 'p",lamdaval.tl,"_%diag%.png'"/;

        put 'plot NaN notitle'/;
        putclose;

        execute 'wgnuplot figure.plt';
);
```

GNUPLOT Produces PNG Files

Generate an output png file labelled $p[\lambda]$, with values of λ ranging from 0 to 10:

```
reset
set grid
set terminal pngcairo size 1209,764 enhanced font 'Verdana,8'
set output 'p10.png'
set xrange [0:8]
set yrange [0:8]
set object 1 rect from 0,0 to  8,8
set arrow from 0,0 to 0,1 nohead ls 0
set arrow from 0,0 to 1,0 nohead ls 0
set arrow from 0,0 to 1,1 nohead ls 0
...
```

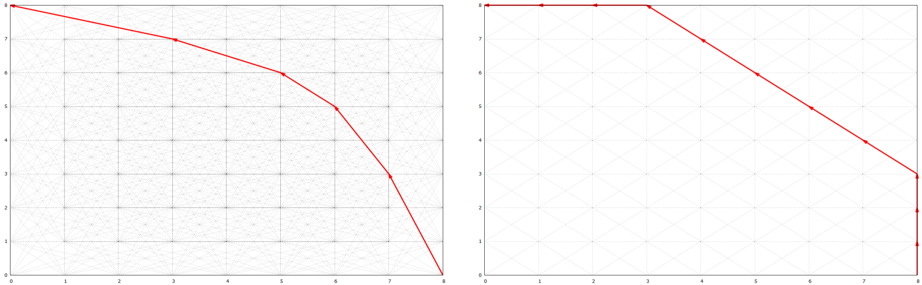
One arrow for each arc in the network, no arrow head, and line style $= 0$ (dashed, light gray).

GNUPLOT Produces PNG Files (cont.)

Complete the figure with arrows indicating arcs on the chosen route (displayed with line style = 1 (solid), line width = 3, and line color red).

```
set arrow from 1,7 to 0,8 head ls 1 lw 3 lc rgb "red"
set arrow from 2,6 to 1,7 head ls 1 lw 3 lc rgb "red"
set arrow from 3,5 to 2,6 head ls 1 lw 3 lc rgb "red"
set arrow from 4,4 to 3,5 head ls 1 lw 3 lc rgb "red"
set arrow from 5,3 to 4,4 head ls 1 lw 3 lc rgb "red"
set arrow from 6,2 to 5,3 head ls 1 lw 3 lc rgb "red"
set arrow from 7,1 to 6,2 head ls 1 lw 3 lc rgb "red"
set arrow from 8,0 to 7,1 head ls 1 lw 3 lc rgb "red"
plot NaN notitle
```

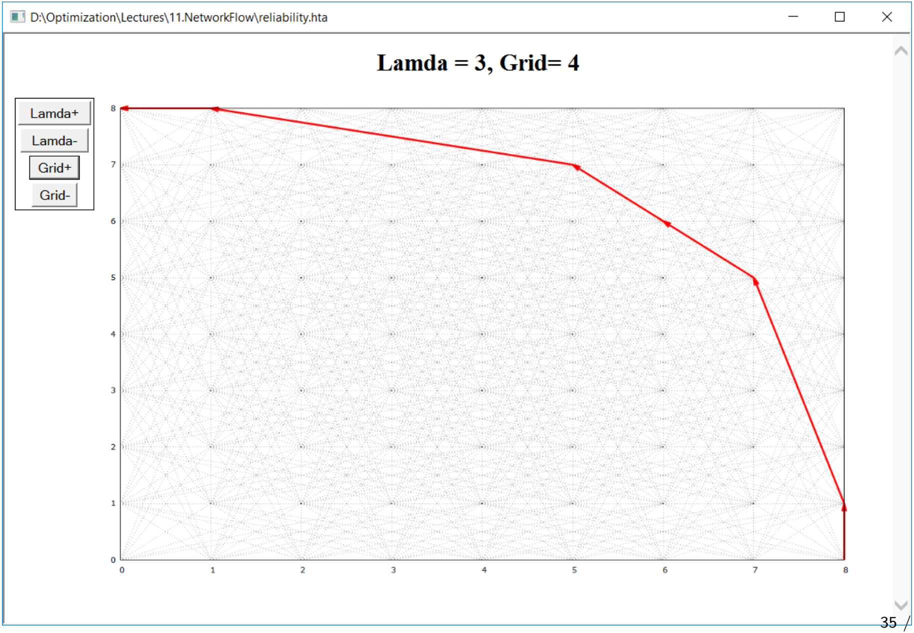
With a Linear Objective, Discretization Matters ($\lambda = 0.4$)



A Few Lines of Javascript Produces `figure.hta`

```
<!DOCTYPE html>
<html><head><script>
var lamda = 0;  var grid = 1;
function Path(v)
  lamda = lamda + v;  lamda = Math.max(0,lamda);  lamda = Math.min(10,lamda);
  document.getElementById("map").src = './p' + lamda + '_' + grid + '.png';
  document.getElementById("title").innerHTML = "<h2>Lamda = " + lamda + ", Grid= " + grid + "</h2>";
function Grid(v)
  grid = grid + v; grid = Math.max(1,grid); grid = Math.min(4,grid);
  document.getElementById("map").src = './p' + lamda + '_' + grid + '.png';
  document.getElementById("title").innerHTML = "<h2>Lamda = " + lamda + ", Grid= " + grid + "</h2>";
</script></head>

<table>
<tr><td></td><td id="title" align="center"><h2>Lamda = 0, Grid=1</h2></td></tr>
<tr><td style="vertical-align:top">
  <table style="border:1px solid black; padding: 5px;">
    <tr><td align="center"><button type="button" onclick="Path(1)">Lamda+</button></td></tr>
    <tr><td align="center"><button type="button" onclick="Path(-1)">Lamda-</button></td></tr>
    <tr><td align="center"><button type="button" onclick="Grid(1)">Grid+</button></td></tr>
    <tr><td align="center"><button type="button" onclick="Grid(-1)">Grid-</button></td></tr>
  </table></td>
      <th rowspan="2"><img id="map" src="./p0_1.png" style="height:600px"></th></tr>
</body>
</html>
```

HTA Interface for Reporting

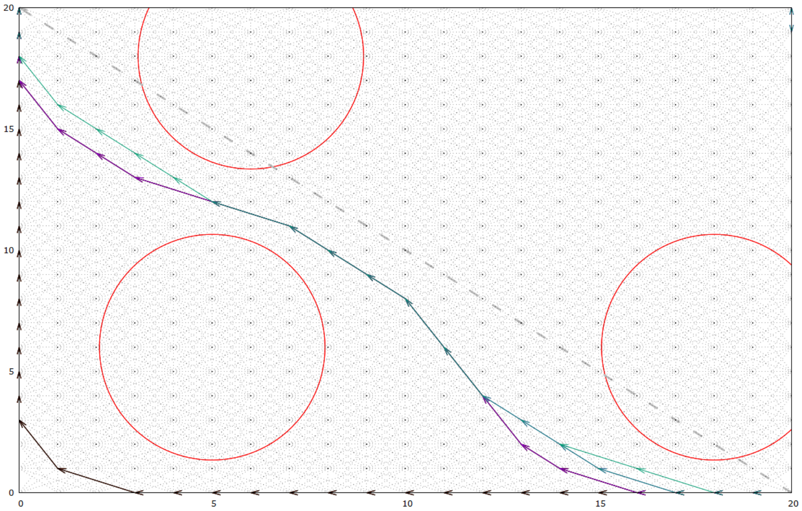


Risk-Contrained Route Choice: GAMS

```
parameter       rmax    Maximum tolerable risk;

variable        OBJ     Objective function;

nonnegative
variables       X               Route choice;

equations       conservation, risktol, objdef;

objdef..
                OBJ =e= sum(a, X(a) * dist(a));
risktol..

        log(rmax) =g= sum(a, X(a) * dist(a) * log(p(a)));

conservation(k)..

        sum(a(i,k), X(a)) + 1$start(k) =e= sum(a(k,j),X(a)) + 1$end(k);

model routechoice /conservation, risktol, objdef/;

rmax = pref;
solve routechoice using lp minimizing obj;
```

Risk-Constrained Route Choice: The Dual

```
variable        OBJD    Dual Objective,
                PRISK   Shadow price on the risk constraint,
                T       Time to reach endpoint from given node;

equations       dualobj Defines OBJDUAL
                dualcon Defines dual constraints;

dualobj..

        OBJD =e= sum(end, T(end)) - sum(start,T(start)) + log(rmax)*PRISK;

dualcon(a(i,j))..

        T(j) + dist(a) - PRISK * dist(a)*log(p(a)) =g= T(i);

model dual /dualobj, dualcon/;

PRISK.L = risktol.M;
T.L(k) = conservation.M(k);
OBJD.L = OBJ.L;
solve dual using lp maximizing OBJD;
```



0
5
10
15
20