CS270: Lecture 2.

Admin:

CS270: Lecture 2.

Admin:
Check Piazza.

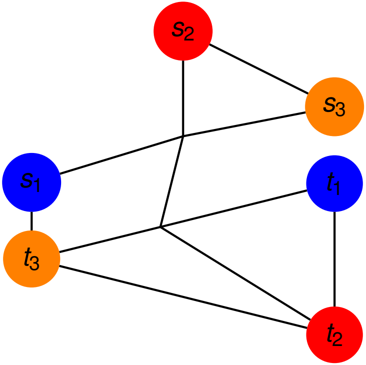
CS270: Lecture 2.

Admin:
Check Piazza.
Today:

- Finish Path Routing.
- ????

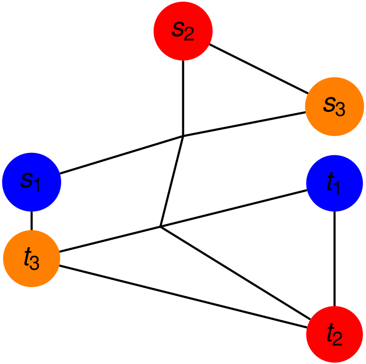
Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



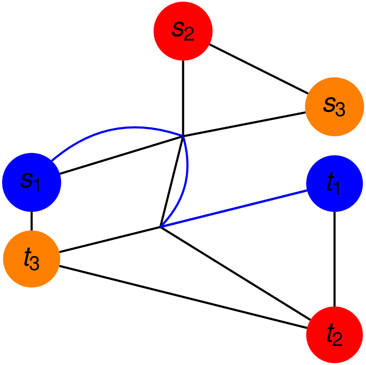
Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



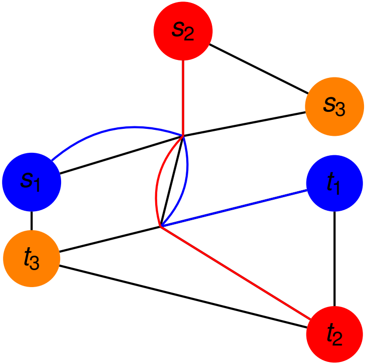
Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



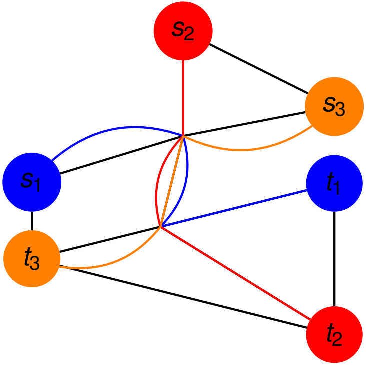
Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



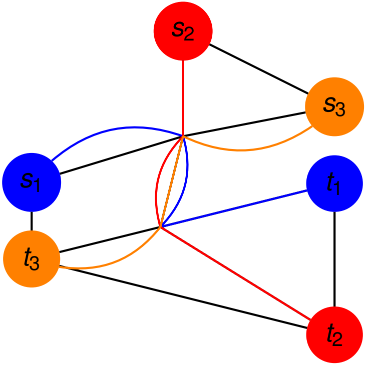
Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



Path Routing.

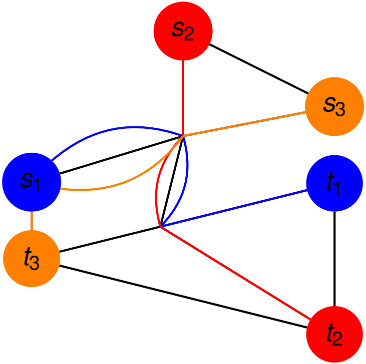
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



Value: 3

Path Routing.

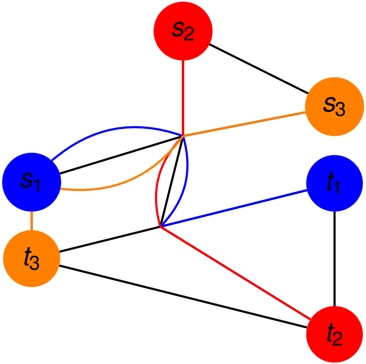
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



Value: 3

Path Routing.

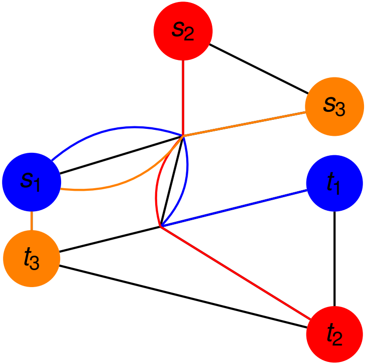
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



~~Value: 3~~

Path Routing.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths connecting s_i and t_i and minimize max load on any edge.



~~Value: 3~~

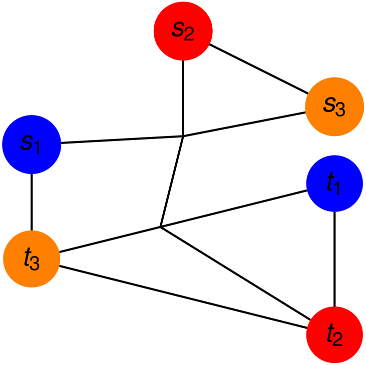
Value: 2

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.

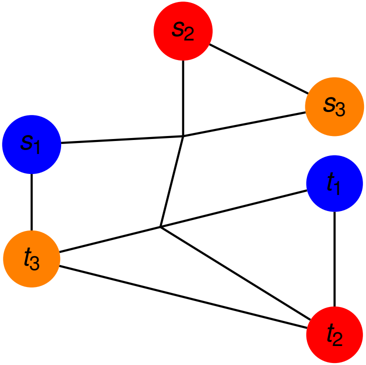
Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



Another problem.

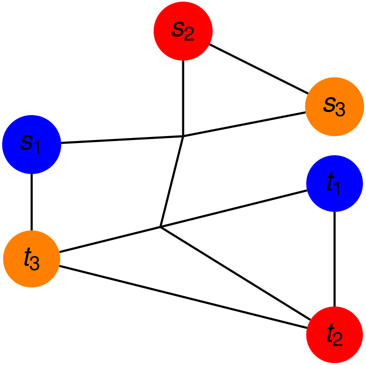
Given $G = (V,E)$, $(s_1,t_1),\ldots,(s_k,t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.

Another problem.

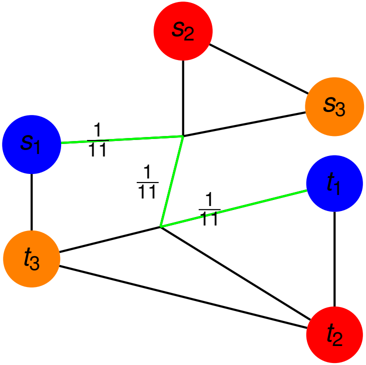
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid:

Another problem.

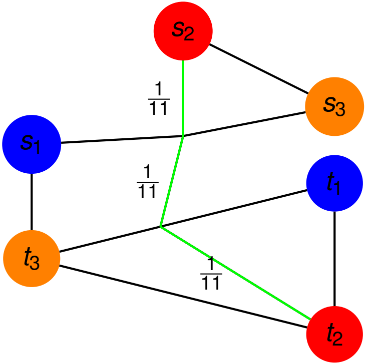
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid: $\frac{3}{11}$

Another problem.

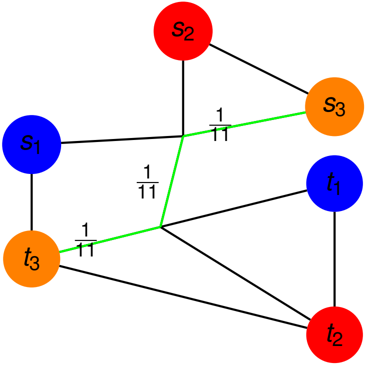
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid: $\frac{3}{11} \quad + \frac{3}{11}$

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.

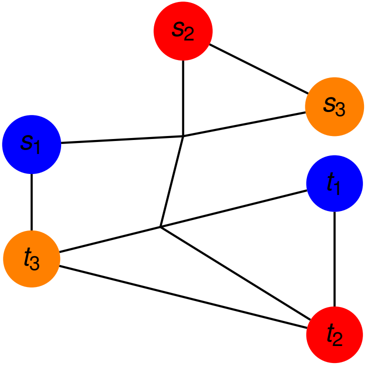


Assign $\frac{1}{11}$ on each of 11 edges.

Toll paid: $\frac{3}{11} \quad + \frac{3}{11} \ + \frac{3}{11}$

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.

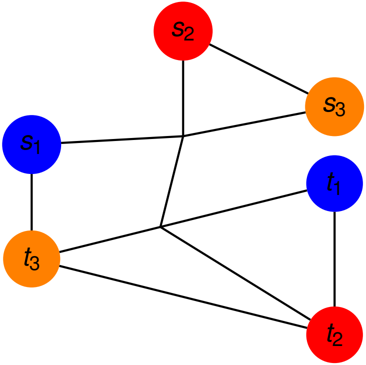


Assign $\frac{1}{11}$ on each of 11 edges.

Toll paid: $\frac{3}{11} \quad + \frac{3}{11} \;+ \frac{3}{11} = \frac{9}{11}$

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



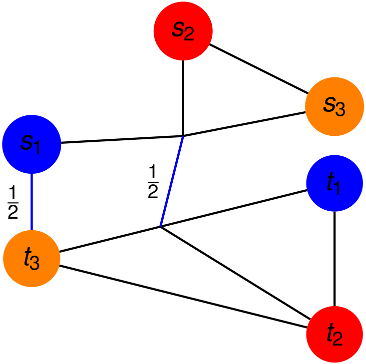
Assign $\frac{1}{11}$ on each of 11 edges.

Toll paid: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.

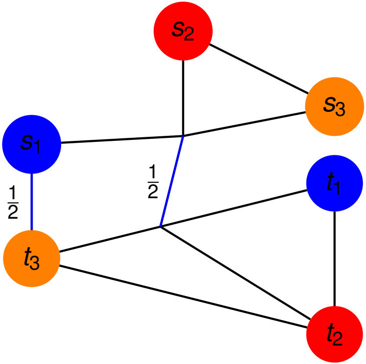
Toll paid: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



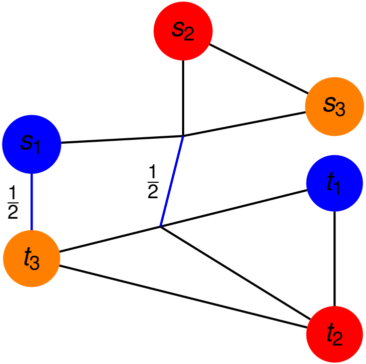
Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Toll paid:

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \dots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



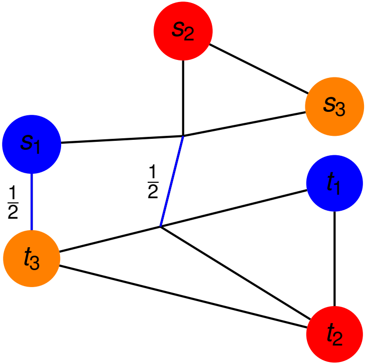
Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid: $\frac{3}{11} \quad + \frac{3}{11} \ + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Toll paid: $\frac{1}{2} + \frac{1}{2} + \frac{1}{2}$

Another problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Toll paid: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Toll paid: $\frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{3}{2}$

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Congestion of edge, e: $c(e)$

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Congestion of edge, e: $c(e)$
number of paths in routing that contain e.

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Congestion of edge, e: $c(e)$
number of paths in routing that contain e.

Congestion of routing:

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Congestion of edge, e: $c(e)$
number of paths in routing that contain e.

Congestion of routing: maximum congestion of any edge.

Terminology

Routing: Paths $p_1, p_2, \ldots, p_k$, p_i connects s_i and t_i.

Congestion of edge, e: $c(e)$
number of paths in routing that contain e.

Congestion of routing: maximum congestion of any edge.

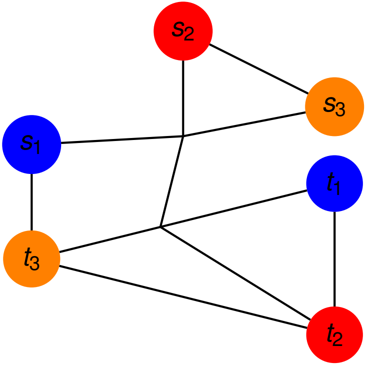
Find routing that minimizes congestion (or maximum congestion.)

Toll problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.

Toll problem.

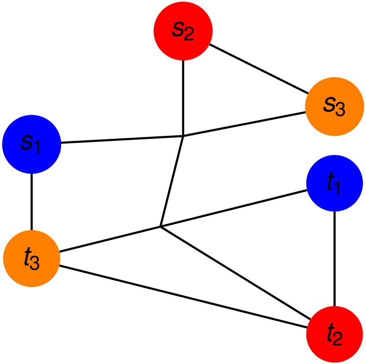
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.

Toll problem.

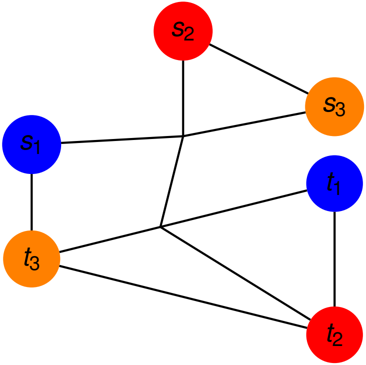
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Total toll:

Toll problem.

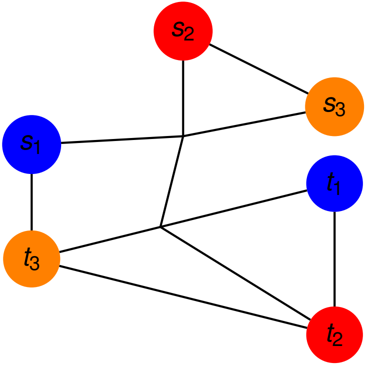
Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11}$

Toll problem.

Given $G=(V,E)$, $(s_1,t_1),\ldots,(s_k,t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.

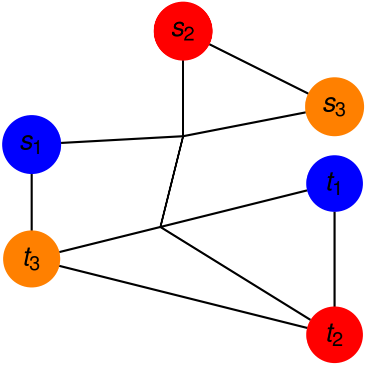


Assign $\frac{1}{11}$ on each of 11 edges.

Total toll: $\frac{3}{11}+\frac{3}{11}+\frac{3}{11}=\frac{9}{11}$

Toll problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



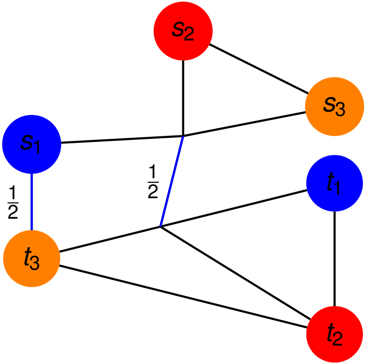
Assign $\frac{1}{11}$ on each of 11 edges.

Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Toll problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of “toll” to edges to maximize total toll for connecting pairs.



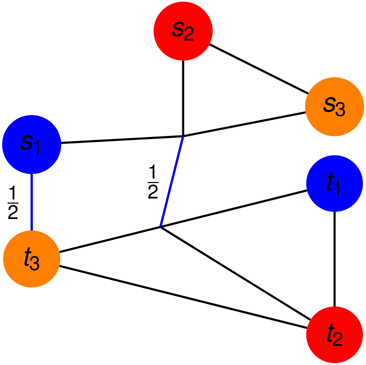
Assign $\frac{1}{11}$ on each of 11 edges.
Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.

Toll problem.

Given $G = (V,E)$, $(s_1,t_1),\ldots,(s_k,t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



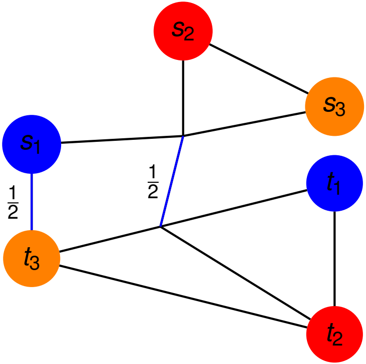
Assign $\frac{1}{11}$ on each of 11 edges.
Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Total toll:

Toll problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



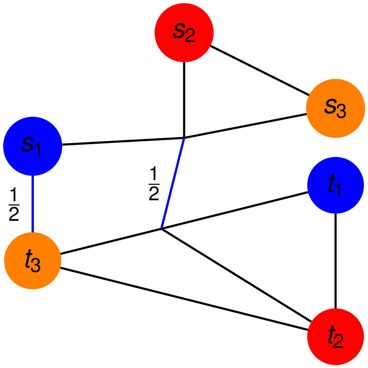
Assign $\frac{1}{11}$ on each of 11 edges.
Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Total toll: $\frac{1}{2} + \frac{1}{2} + \frac{1}{2}$

Toll problem.

Given $G = (V, E)$, $(s_1, t_1), \ldots, (s_k, t_k)$, find a set of k paths assign one unit of "toll" to edges to maximize total toll for connecting pairs.



Assign $\frac{1}{11}$ on each of 11 edges.
Total toll: $\frac{3}{11} + \frac{3}{11} + \frac{3}{11} = \frac{9}{11}$

Can we do better?

Assign $1/2$ on these two edges.
Total toll: $\frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{3}{2}$

Toll: Terminology.

$d(e)$ - toll assigned to edge e.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.
$d(u,v)$ - total assigned to shortest path between u and v.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.
$d(u,v)$ - total assigned to shortest path between u and v.
$d(x)$ - polymorpic

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.
$d(u,v)$ - total assigned to shortest path between u and v.
$d(x)$ - polymorpic
  x could be edge, path, or pair.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.
$d(u,v)$ - total assigned to shortest path between u and v.
$d(x)$ - ~~polymorpic~~
  x could be edge, path, or pair.

Toll: Terminology.

$d(e)$ - toll assigned to edge e.
Note: $\sum_e d(e) = 1$. $d(p)$ - total toll assigned to path p.
$d(u,v)$ - total assigned to shortest path between u and v.
$d(x)$ - ~~polymorpic~~ polymorphic
x could be edge, path, or pair.

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:

$\max_e c(e) \geq \sum_e c(e)d(e)$

Total length is total congestion:

$\sum_e c(e)d(e) = \sum_i d(p_i)$

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

$$\max_e c(e) \geq \sum_e c(e)d(e) = \sum_i d(p_i) \geq \sum_i d(s_i, t_i).$$

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

$$\max_e c(e) \geq \sum_e c(e)d(e) = \sum_i d(p_i) \geq \sum_i d(s_i, t_i).$$

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

$$\max_e c(e) \geq \sum_e c(e)d(e) = \sum_i d(p_i) \geq \sum_i d(s_i, t_i).$$

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

$$\max_e c(e) \geq \sum_e c(e)d(e) = \sum_i d(p_i) \geq \sum_i d(s_i, t_i).$$

A toll solution is lower bound on any routing solution.

Toll is lower bound on Path Routing.

From before:
Max bigger than minimum weighted average:
$\max_e c(e) \geq \sum_e c(e)d(e)$
Total length is total congestion:
$\sum_e c(e)d(e) = \sum_i d(p_i)$

Each path, p_i, in routing has length $d(p_i) \geq d(s_i, t_i)$.

$$\max_e c(e) \geq \sum_e c(e)d(e) = \sum_i d(p_i) \geq \sum_i d(s_i, t_i).$$

A toll solution is lower bound on any routing solution.

Any routing solution is an upper bound on a toll solution.

Algorithm.

Assign tolls according to routing.

Algorithm.

Assign tolls according to routing.

How to route?

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls?

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:
The shortest path routing has has $d(e) \propto 2^{c(e)}$.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:
The shortest path routing has has $d(e) \propto 2^{c(e)}$.

"The routing is stable, the tolls are stable."

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:
The shortest path routing has has $d(e) \propto 2^{c(e)}$.

"The routing is stable, the tolls are stable."

Routing: each path p_i in routing is a shortest path w.r.t $d(\cdot)$

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:
The shortest path routing has <u>has</u> $d(e) \propto 2^{c(e)}$.

“The routing is stable, the tolls are stable.”

Routing: each path p_i in routing is a shortest path w.r.t $d(\cdot)$

Tolls: ...where $d(e)$ is defined w.r.t. to current routing.

Algorithm.

Assign tolls according to routing.

How to route? Shortest paths!

Assign routing according to tolls.

How to assign tolls? Higher tolls on congested edges.

Toll: $d(e) =\propto 2^{c(e)}$. $\sum_e d(e) = 1$.

Equilibrium:
The shortest path routing has <u>has</u> $d(e) \propto 2^{c(e)}$.

“The routing is stable, the tolls are stable.”

Routing: each path p_i in routing is a shortest path w.r.t $d(\cdot)$

Tolls: ...where $d(e)$ is defined w.r.t. to current routing.
Subtlety here due to $\sum_e d(e) = 1$.

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$c_{opt} \geq \sum_i d(s_i, t_i) = \sum_e d(e)c(e)$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$\begin{aligned} c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e) c(e) \\ &= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) \end{aligned}$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$
\begin{aligned}
c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e) c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}}
\end{aligned}
$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$
\begin{aligned}
c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e)c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}}
\end{aligned}
$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$
\begin{aligned}
c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e)c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \quad \text{Let } c_t = c_{max} - 2\log m. \\
&\geq \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\leq c_t} 2^{c(e)}}
\end{aligned}
$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

$$
\begin{aligned}
c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e)c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \\
&\geq \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\leq c_t} 2^{c(e)}}
\end{aligned}
$$

Let $c_t = c_{max} - 2\log m$.

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

For e with $c(e) \leq c_{max} - 2\log m$; $2^{c(e)} \leq 2^{c_{\max} - 2\log m} = \frac{2^{c_{max}}}{m^2}$.

$$
\begin{aligned}
c_{opt} &\geq \sum_i d(s_i, t_i) = \sum_e d(e)c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \quad \text{Let } c_t = c_{max} - 2\log m. \\
&\geq \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\leq c_t} 2^{c(e)}} \\
&\geq \frac{(c_t) \sum_{e:c(e)>c_t} 2^{c(e)}}{(1 + \frac{1}{m}) \sum_{e:c(e)>c_t} 2^{c(e)}}
\end{aligned}
$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

For e with $c(e) \le c_{max} - 2\log m$; $2^{c(e)} \le 2^{c_{\max} - 2\log m} = \frac{2^{c_{max}}}{m^2}$.

$$
\begin{aligned}
c_{opt} &\ge \sum_i d(s_i, t_i) = \sum_e d(e) c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \quad \text{Let } c_t = c_{max} - 2\log m. \\
&\ge \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\le c_t} 2^{c(e)}} \\
&\ge \frac{(c_t) \sum_{e:c(e)>c_t} 2^{c(e)}}{(1+\frac{1}{m}) \sum_{e:c(e)>c_t} 2^{c(e)}} \\
&\ge \frac{(c_t)}{1+\frac{1}{m}} = \frac{c_{\max} - 2\log m}{(1+\frac{1}{m})}
\end{aligned}
$$

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

For e with $c(e) \le c_{max} - 2\log m$; $2^{c(e)} \le 2^{c_{\max} - 2\log m} = \frac{2^{c_{max}}}{m^2}$.

$$
\begin{aligned}
c_{opt} &\ge \sum_i d(s_i, t_i) = \sum_e d(e)c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \quad \text{Let } c_t = c_{max} - 2\log m. \\
&\ge \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\le c_t} 2^{c(e)}} \\
&\ge \frac{(c_t) \sum_{e:c(e)>c_t} 2^{c(e)}}{(1+\frac{1}{m}) \sum_{e:c(e)>c_t} 2^{c(e)}} \\
&\ge \frac{(c_t)}{1+\frac{1}{m}} = \frac{c_{\max} - 2\log m}{(1+\frac{1}{m})}
\end{aligned}
$$

Or $c_{max} \le (1+\frac{1}{m}) c_{opt} + 2\log m$.

How good is equilibrium?

Path is routed along shortest path and $d(e) = \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}}$.

For e with $c(e) \le c_{max} - 2\log m$; $2^{c(e)} \le 2^{c_{\max} - 2\log m} = \frac{2^{c_{max}}}{m^2}$.

$$
\begin{aligned}
c_{opt} &\ge \sum_i d(s_i, t_i) = \sum_e d(e) c(e) \\
&= \sum_e \frac{2^{c(e)}}{\sum_{e'} 2^{c(e')}} c(e) = \frac{\sum_e 2^{c(e)} c(e)}{\sum_e 2^{c(e)}} \quad \text{Let } c_t = c_{max} - 2\log m. \\
&\ge \frac{\sum_{e:c(e)>c_t} 2^{c(e)} c(e)}{\sum_{e:c(e)>c_t} 2^{c(e)} + \sum_{e:c(e)\le c_t} 2^{c(e)}} \\
&\ge \frac{(c_t) \sum_{e:c(e)>c_t} 2^{c(e)}}{(1 + \frac{1}{m}) \sum_{e:c(e)>c_t} 2^{c(e)}} \\
&\ge \frac{(c_t)}{1 + \frac{1}{m}} = \frac{c_{\max} - 2\log m}{(1 + \frac{1}{m})}
\end{aligned}
$$

Or $c_{max} \le (1 + \frac{1}{m}) c_{opt} + 2\log m$.

(Almost) within additive term of $2\log m$ of optimal!

Getting to equilibrium.

Maybe no equilibrium!

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length
within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length
within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.
$c_{opt} \geq \sum_i d(s_i, t_i) \geq \frac{1}{3} \sum_e d(p_i)$

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.
$c_{opt} \geq \sum_i d(s_i, t_i) \geq \frac{1}{3} \sum_e d(p_i) = \frac{1}{3} \sum_e d(e)c(e)$

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.
$c_{opt} \geq \sum_i d(s_i, t_i) \geq \frac{1}{3} \sum_e d(p_i) = \frac{1}{3} \sum_e d(e) c(e)$

We obtain $c_{max} = 3(1 + \frac{1}{m}) c_{opt} + 2 \log m$.

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length
within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.
$c_{opt} \geq \sum_i d(s_i, t_i) \geq \frac{1}{3} \sum_e d(p_i) = \frac{1}{3} \sum_e d(e)c(e)$

We obtain $c_{max} = 3(1 + \frac{1}{m})c_{opt} + 2\log m$.

This is worse!

Getting to equilibrium.

Maybe no equilibrium!

Approximate equilibrium:

Each path is routed along a path with length
within a factor of 3 of the shortest path and $d(e) \propto 2^{c(e)}$.

Lose a factor of three at the beginning.
$c_{opt} \geq \sum_i d(s_i, t_i) \geq \frac{1}{3} \sum_e d(p_i) = \frac{1}{3} \sum_e d(e) c(e)$

We obtain $c_{max} = 3(1 + \frac{1}{m}) c_{opt} + 2 \log m$.

This is worse!
What do we gain?

An algorithm!

An algorithm!

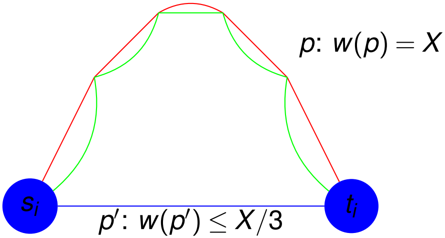
Repeat: reroute any path that is off by a factor of 3.

An algorithm!

Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)

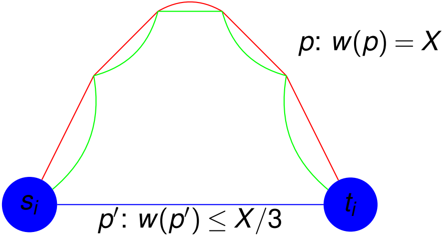
An algorithm!

Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



An algorithm!

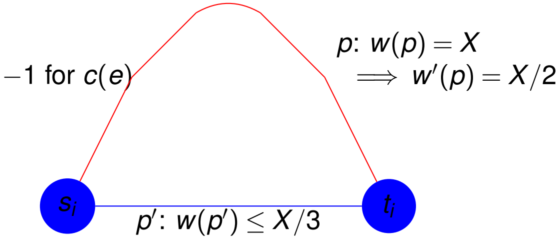
Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$

An algorithm!

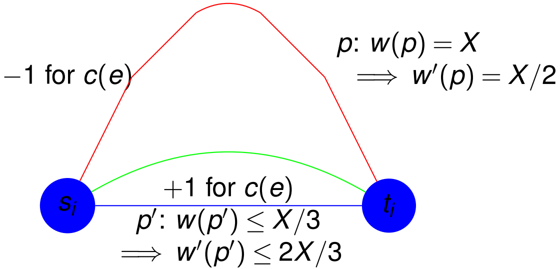
Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:

An algorithm!

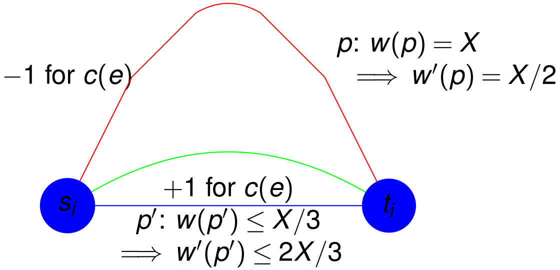
Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:
Divides $w(e)$ along long path (with $w(p)$ of X) by two.

An algorithm!

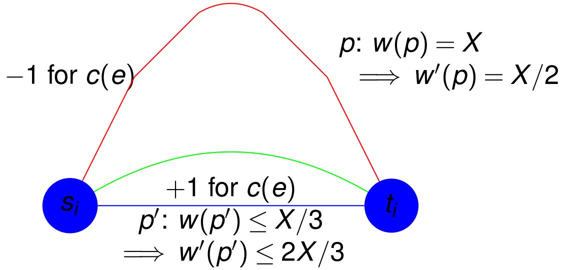
Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:
Divides $w(e)$ along long path (with $w(p)$ of X) by two.
Multiplies $w(e)$ along shorter ($w(p) \leq X/3$) path by two.

An algorithm!

Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)

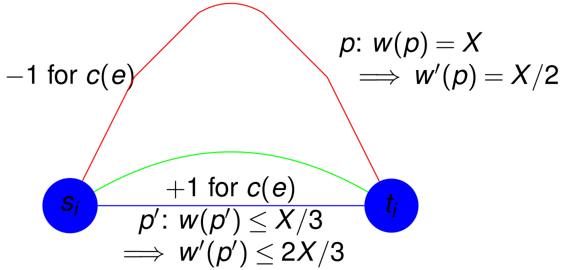


Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:
Divides $w(e)$ along long path (with $w(p)$ of X) by two.
Multiplies $w(e)$ along shorter ($w(p) \le X/3$) path by two.

$$-\frac{X}{2} + \frac{X}{3} = -\frac{X}{6}.$$

An algorithm!

Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



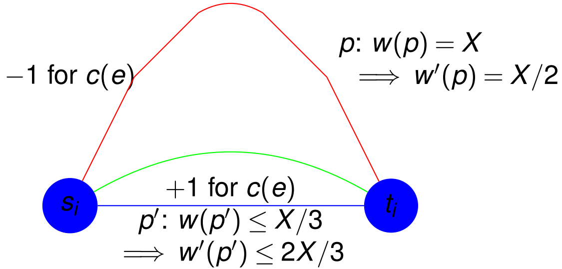
Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:
Divides $w(e)$ along long path (with $w(p)$ of X) by two.
Multiplies $w(e)$ along shorter ($w(p) \leq X/3$) path by two.

$$-\frac{X}{2} + \frac{X}{3} = -\frac{X}{6}.$$

Potential function decreases.

An algorithm!

Repeat: reroute any path that is off by a factor of 3.
(Note: $d(e)$ recomputed every rerouting.)



Potential function: $\sum_e w(e)$, $w(e) = 2^{c(e)}$
Moving path:
Divides $w(e)$ along long path (with $w(p)$ of X) by two.
Multiplies $w(e)$ along shorter ($w(p) \leq X/3$) path by two.

$$-\frac{X}{2} + \frac{X}{3} = -\frac{X}{6}.$$

Potential function decreases. $\implies$ termination and existence.

Tuning...

Tuning...

Replace $d(e) = (1 + \varepsilon)^{c(e)}$.

Tuning...

Replace $d(e) = (1+\varepsilon)^{c(e)}$.

Replace factor of 3 by $(1+2\varepsilon)$

Tuning...

Replace $d(e) = (1+\varepsilon)^{c(e)}$.

Replace factor of 3 by $(1+2\varepsilon)$

$c_{max} \leq (1+2\varepsilon)c_{opt} + 2\log m/\varepsilon$.. (Roughly)

Tuning...

Replace $d(e) = (1+\varepsilon)^{c(e)}$.

Replace factor of 3 by $(1+2\varepsilon)$

$c_{max} \leq (1+2\varepsilon)c_{opt} + 2\log m/\varepsilon$.. (Roughly)

Fractional paths?

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on congested edges.

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on congested edges.
Routing player routes on only cheap paths.

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

$$\sum_i d(s_i, t_i) \quad =$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths.

$$\sum_i d(s_i, t_i) \quad = \quad \sum_i d(p_i)$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.

Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$$\begin{aligned}\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\ &= \sum_e c(e) d(e)\end{aligned}$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$.

$$\begin{aligned}\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\ &= \sum_e c(e)d(e) \\ &= \sum_{e:d(e)>0} c(e)d(e)\end{aligned}$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i,t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion.

$$\begin{aligned}
\sum_i d(s_i,t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e))
\end{aligned}$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i,t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i,t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

Any routing solution value $\geq$ **Any** toll solution value.

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$
\begin{aligned}
\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e) d(e) \\
&= \sum_{e: d(e) > 0} c(e) d(e) \\
&= \sum_{e: d(e) > 0} d(e) (\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}
$$

Any routing solution value $\geq$ **Any** toll solution value.
Both these solutions are optimal!!!!!

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

Any routing solution value $\geq$ **Any** toll solution value.
Both these solutions are optimal!!!!! Complementary slackness.

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i,t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i,t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

Any routing solution value $\geq$ **Any** toll solution value.

Both these solutions are optimal!!!!! Complementary slackness.

Why all the mess before?

Revisit Equilibrium.

Solution Pair: $(\{p_i\}, d())$.

Toll Solution Value: $\sum_i d(s_i, t_i)$. Path Routing Value: $\max_e c(e)$.

Toll player assigns toll on only maximally congested edges.
Routing player routes on only cheapest paths.

Routing R uses shortest paths. Summation Switch

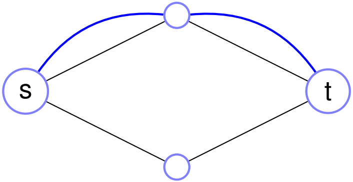
$d(e) \geq 0$. Only Toll on max congestion. $\sum_e d(e) = 1$

$$\begin{aligned}
\sum_i d(s_i, t_i) &= \sum_i d(p_i) \\
&= \sum_e c(e)d(e) \\
&= \sum_{e:d(e)>0} c(e)d(e) \\
&= \sum_{e:d(e)>0} d(e)(\max_e c(e)) \\
&= \max_e c(e)
\end{aligned}$$

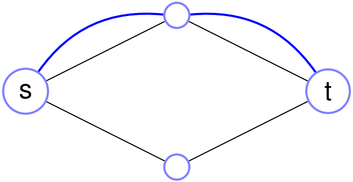
Any routing solution value $\geq$ **Any** toll solution value.
Both these solutions are optimal!!!!! Complementary slackness.

Why all the mess before? To get an algorithm!

Algorithm:exact?

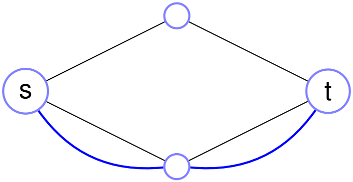


Algorithm:exact?



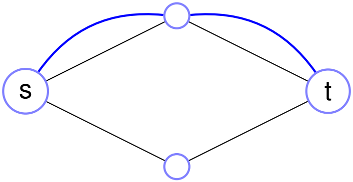
Not shortest when tolls on top.

Algorithm:exact?



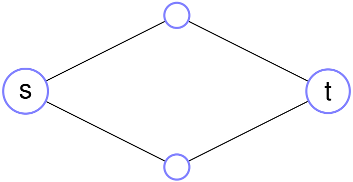
Not shortest when tolls on top.
Hmmm...

Algorithm:exact?



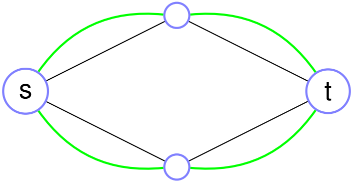
Not shortest when tolls on top.
Hmmm...
Uh oh?

Algorithm:exact?



Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

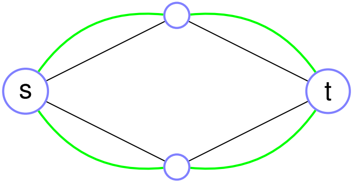
Algorithm:exact?



Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

Hey! Fractional!

Algorithm:exact?

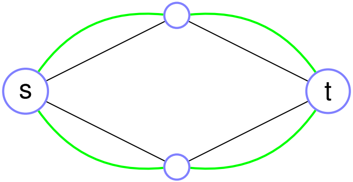


Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

Hey! Fractional!

Use previous algorithms but route two paths between each pair.

Algorithm:exact?



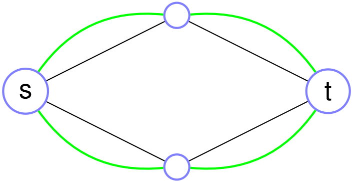
Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

Hey! Fractional!

Use previous algorithms but route two paths between each pair.

Half integral!

Algorithm:exact?



Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

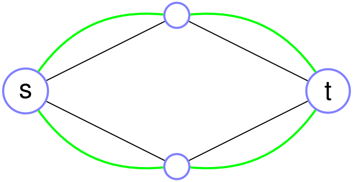
Hey! Fractional!

Use previous algorithms but route two paths between each pair.

Half integral!

Optimality: $(3)C_{\max} + 2\log m/2$.

Algorithm:exact?



Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

Hey! Fractional!

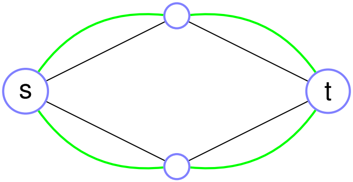
Use previous algorithms but route two paths between each pair.

Half integral!

Optimality: $(3)C_{\max} + 2\log m/2$.

Additive factor shrinking!

Algorithm:exact?



Not shortest when tolls on top.
Hmmm...
Uh oh?
Route half a unit on both!

Hey! Fractional!

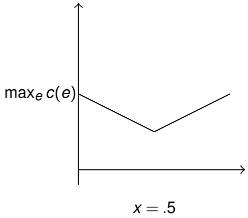
Use previous algorithms but route two paths between each pair.

Half integral!

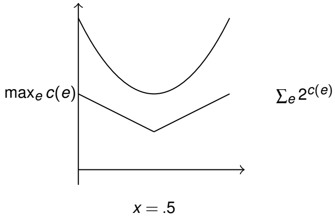
Optimality: $(3)C_{\max} + 2\log m/2$.

Additive factor shrinking!
The 3 can be made $(1+\varepsilon)$ using different base!

Geometrical view.

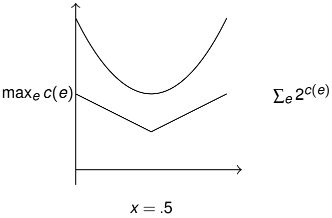


Geometrical view.



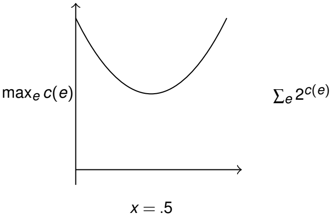
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.

Geometrical view.



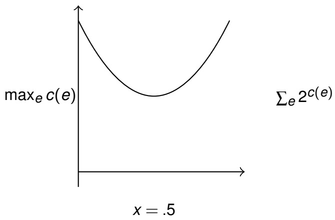
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.

Geometrical view.



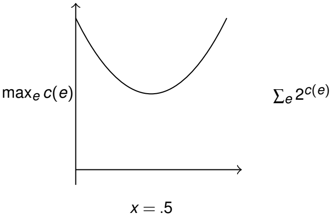
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.

Geometrical view.



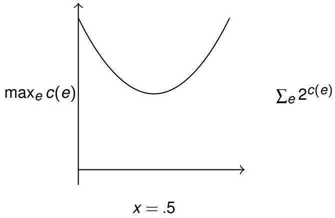
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.

Geometrical view.



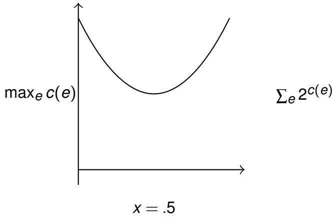
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1.

Geometrical view.



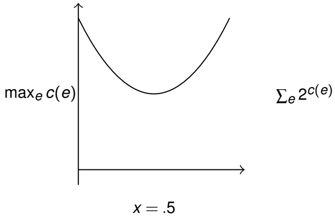
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!

Geometrical view.



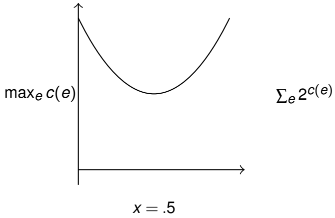
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!

Geometrical view.



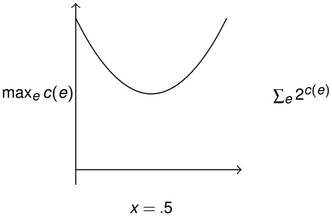
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!
Stepsize=.5.

Geometrical view.



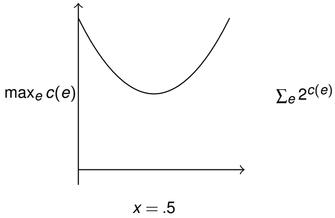
Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!
Stepsize=.5. Back and forth

Geometrical view.



Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!
Stepsize=.5. Back and forth ...but closer to minimum.

Geometrical view.



Smooth: use $\sum_e 2^{c(e)}$ as a proxy for $\max_e c(e)$.
Minimize new function.
Gradient descent.
Stepsize=1. Back and forth!
Stepsize=.5. Back and forth ...but closer to minimum.

Wrap up.

Dueling players:

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.
Congestion player avoids tolls.

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.
Congestion player avoids tolls.

Converges to near optimal solution!

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.
Congestion player avoids tolls.

Converges to near optimal solution!

A lower bound is “necessary” (natural),

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.
Congestion player avoids tolls.

Converges to near optimal solution!

A lower bound is “necessary” (natural),
and helpful (mysterious?)!

Wrap up.

Dueling players:
Toll player raises tolls on congested edges.
Congestion player avoids tolls.

Converges to near optimal solution!

A lower bound is “necessary” (natural),
and helpful (mysterious?)!

Geometric View: Smooth. Gradient Descent. Stepsize.