6.006	Lecture 13	Oct. 27, 2016

TODAY: Shortest Paths Intro.
- shortest paths problem
- negative-weight cycles
- relaxation algorithm/framework
- triangle inequality
- optimal substructure

Motivation:
- Google Maps (fastest/shortest way $A \to B$)
- network routing (fastest way to route packet)

Edge-weighted graph = directed graph $G=(V,E)$
+ edge-weight function $w: E \to \mathbb{R}$

Path $p = v_1 \to v_2 \to \cdots \to v_k$
i.e. $(v_i, v_{i+1}) \in E$ for $1 \le i < k$
— from v_1 to v_k

Weight $w(p) = w(v_1, v_2) + w(v_2, v_3) + \cdots + w(v_{k-1}, v_k)$

Example: $v_1 \xrightarrow{4} v_2 \xrightarrow{-2} v_3 \xrightarrow{-5} v_4 \xrightarrow{1} v_5$

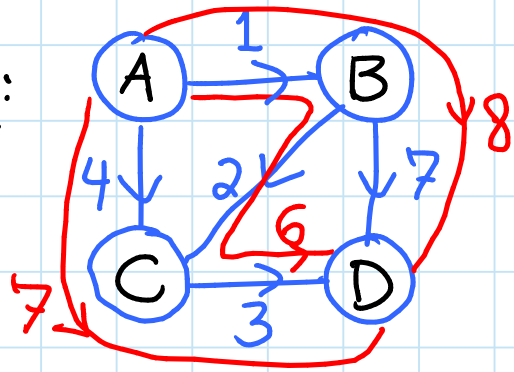
$w(p) = 4 - 2 - 5 + 1 = -2$

Shortest path from $u \in V$ to $v \in V$

= a path p of minimum possible weight $w(p)$ from u to v (among all paths from u to v)

- weight of such a path is shortest-path weight $\delta(u,v)$

Example:



$A \to B \to C \to D$

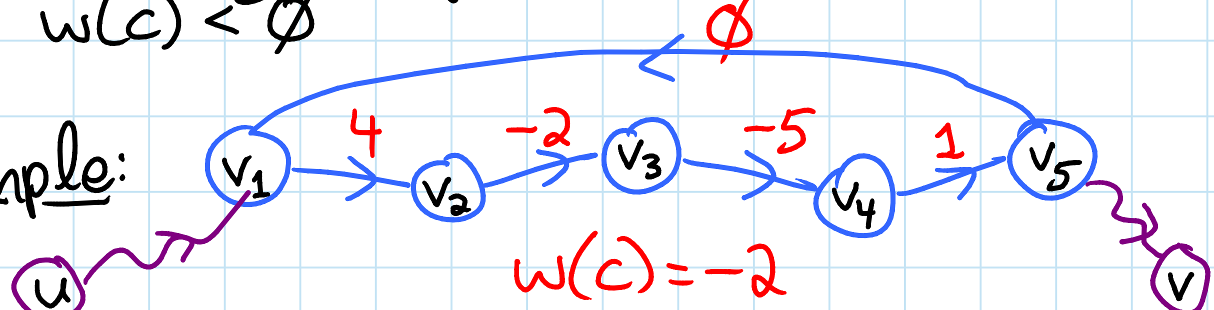
$\delta(A,D) = 6$

- if no path from u to v then no shortest path either
  - define $\delta(u,v) = \infty$

e.g. $\delta(B,A) = \infty$

- even if there's a path from u to v, there may not be a shortest path:
- negative-weight cycle $c = v_1 \to v_2 \to \cdots \to v_k \to v_1$ has $w(c) < \emptyset$

Example:



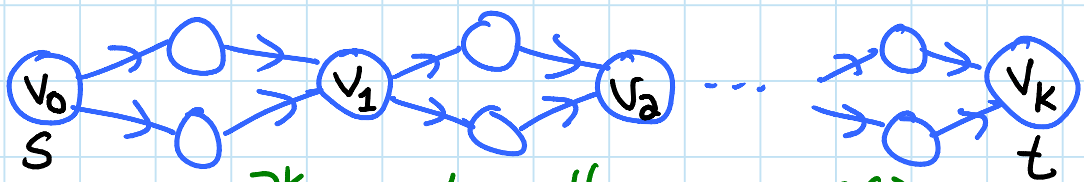
- $\delta(u,v) = \begin{cases} \inf\{w(p) \mid \text{path } p \text{ from } u \text{ to } v\} \\ \infty \text{ if no path from } u \text{ to } v \end{cases}$

  $\Rightarrow -\infty$ if negative-weight cycle along any path from u to v

Brute-force algorithm:

```
distance(s, t):
    for each path p from s to t:
        compute w(p)
    return p encountered with smallest w(p)
```

- infinitely many paths ($\Rightarrow$ infinite time) if $\delta(s, t) = -\infty$ (i.e. neg.-weight cycle on the way)
- even when finite, often exponential # paths:
  - $(n-2)!$ in complete graph
  - $2^{\Theta(n)}$ even in directed acyclic graph (DAG):



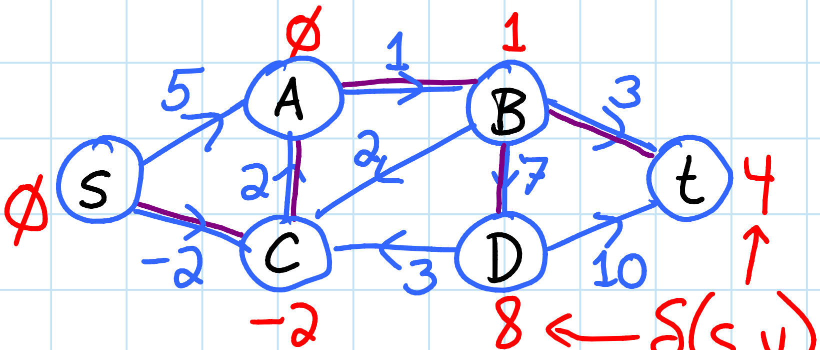
2^k $s \to t$ paths; $n = \Theta(k)$

Single-Source Shortest Paths (SSSP) Problem:

- given edge-weighted directed graph $G=(V,E,w)$
  & source vertex $s \in V$
- compute $\delta(s,v)$ for all $v \in V$
- turns out we can do this just as fast as computing $\delta(s,v)$ for a single $v \in V$

- ideally also compute shortest-path tree containing a shortest path from s to every $v \in V$
  - represent by storing v.parent for each $v \in V$
    ↳ π in textbook

Example:

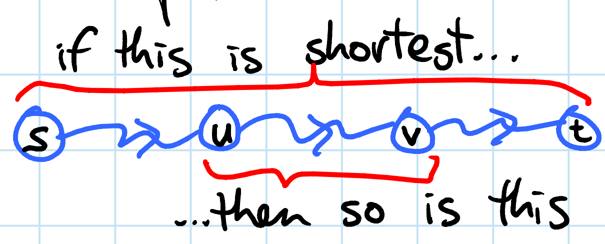


v	v.parent
s	None
A	C
B	A
C	s
D	B
t	B

- why can SSSPs form a tree?

Optimal substructure: subpaths of shortest paths are shortest paths

Proof: cut & paste
- shorter $u \to v$ path would make a shorter $s \to t$ path, contradiction. □



Relaxation algorithm for SSSP:

framework for most shortest-path algorithms

- maintain distance estimate $v.d$ for each $v \in V$
- goal: $v.d = \delta(s,v)$ for all $v \in V$
- invariant: $v.d \geq \delta(s,v)$ for all $v \in V$
- initialization(V):

```
for v in V:
    v.d = ∞
    v.parent = None
s.d = 0
```

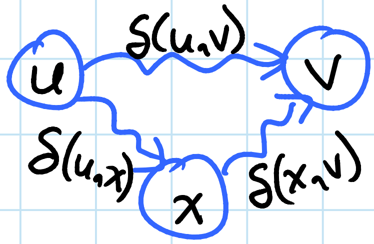
satisfies invariant

- repeatedly improve estimates until reach the goal

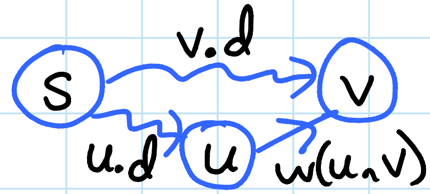
- "relaxation" refers to letting a solution (temporarily) violate a constraint, and trying to fix these violations
- for shortest paths, the constraint is:

Triangle inequality: $\delta(u,v) \leq \delta(u,x) + \delta(x,v)$ for all $u,v,x \in V$

Proof: shortest path $u \to v$
$\leq$ any particular path
e.g. $u \to x \to v$ $\square$



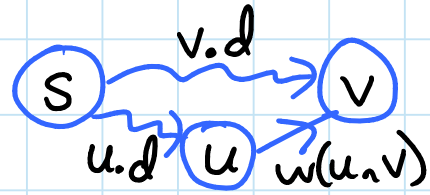
Edge relaxation:



- consider edge (u,v)
- $\delta(s,v) \leq \delta(s,u) + \delta(u,v)$ — Δ inequality
  $\leq \delta(s,u) + w(u,v)$ — candidate path

$\Rightarrow$ want $v.d \leq u.d + w(u,v)$

<u>relax</u> (u,v)




```
if v.d > u.d + w(u,v)
    v.d = u.d + w(u,v)
    v.parent = u
```

Safety Lemma: relax(u,v) maintains invariant $x.d \geq \delta(s,x)$ for all $x \in V$

Proof: only $v.d$ changes, to:

$u.d + w(u,v)$

$\geq \delta(s,u) + w(u,v)$ — invariant before

$\geq \delta(s,u) + \delta(u,v)$ — candidate path

$\geq \delta(s,v)$ $\square$ — Δ inequality

Relaxation SSSP algorithm:

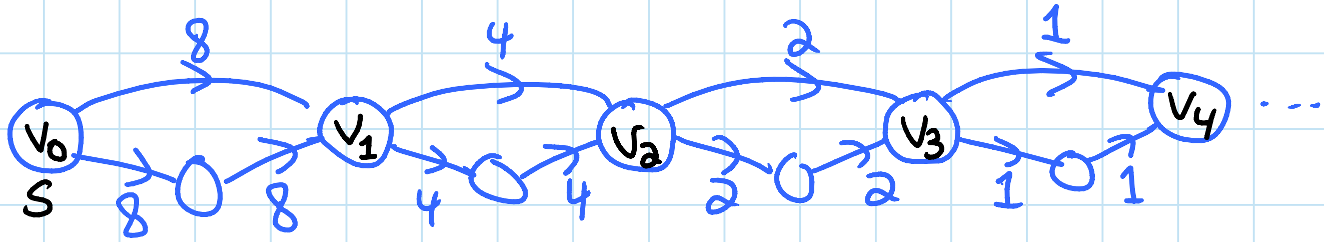
```
initialization(V)
while some edge (u,v) has v.d > u.d + w(u,v):
    pick such an edge (u,v)
    relax(u,v)
```

Half Correctness: $v.d \geq \delta(s,v)$ for all $v \in V$

Proof: by induction on # relaxations
- base: initialization
- step: safeness lemma

Infinite relaxation: whenever negative-weight cycle reachable from source s
- relaxation continues around the cycle forever
- proof: always finite $v.d$'s, but need $-\infty$

Exponential relaxation: even in a DAG

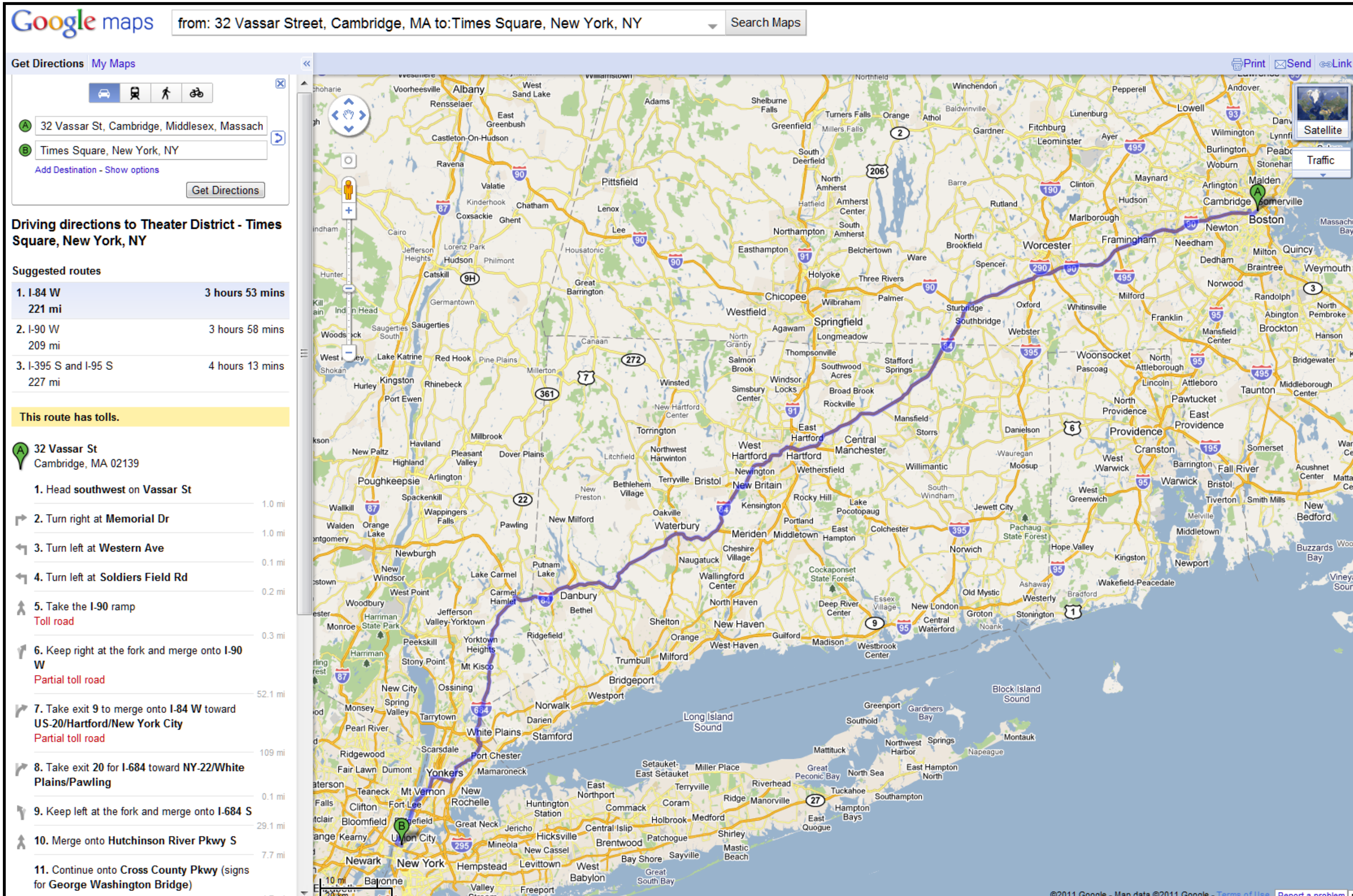


d:	v_0		v_1		v_2		v_3		v_4
	$\emptyset$	∞	∞	∞	∞	∞	∞	∞	∞
		8	16	20	24	26	28	29	30
									29
							26	27	28
									27
					20	22	24	25	26
									25
							22	23	24
									23
					etc.				

relax upper edges right to left
in between each: relax lower edges left to right

NEXT: Bellman-Ford algorithm finds a polynomial-length relaxation order!

Shortest Paths



Network Routing

http://s8.postimg.org/8kp2a7yqd/Client_routing_table.png

IPv4 Route Table

===

Active Routes:

Network Destination	Netmask	Gateway	Interface	Metric
0.0.0.0	0.0.0.0	192.168.3.1	192.168.3.41	281
0.0.0.0	0.0.0.0	10.8.0.1	10.8.0.2	286
10.8.0.0	255.255.255.252	On-link	10.8.0.2	286
10.8.0.2	255.255.255.255	On-link	10.8.0.2	286
10.8.0.3	255.255.255.255	On-link	10.8.0.2	286
127.0.0.0	255.0.0.0	On-link	127.0.0.1	306
127.0.0.1	255.255.255.255	On-link	127.0.0.1	306
127.255.255.255	255.255.255.255	On-link	127.0.0.1	306
169.254.0.0	255.255.0.0	On-link	169.254.112.65	276
169.254.112.65	255.255.255.255	On-link	169.254.112.65	276
169.254.255.255	255.255.255.255	On-link	169.254.112.65	276
192.168.1.0	255.255.255.0	10.8.0.1	10.8.0.2	30
192.168.3.0	255.255.255.0	On-link	192.168.3.41	281
192.168.3.41	255.255.255.255	On-link	192.168.3.41	281
192.168.3.255	255.255.255.255	On-link	192.168.3.41	281
224.0.0.0	240.0.0.0	On-link	127.0.0.1	306
224.0.0.0	240.0.0.0	On-link	169.254.112.65	276
224.0.0.0	240.0.0.0	On-link	10.8.0.2	286
224.0.0.0	240.0.0.0	On-link	192.168.3.41	281
255.255.255.255	255.255.255.255	On-link	127.0.0.1	306
255.255.255.255	255.255.255.255	On-link	169.254.112.65	276
255.255.255.255	255.255.255.255	On-link	10.8.0.2	286
255.255.255.255	255.255.255.255	On-link	192.168.3.41	281

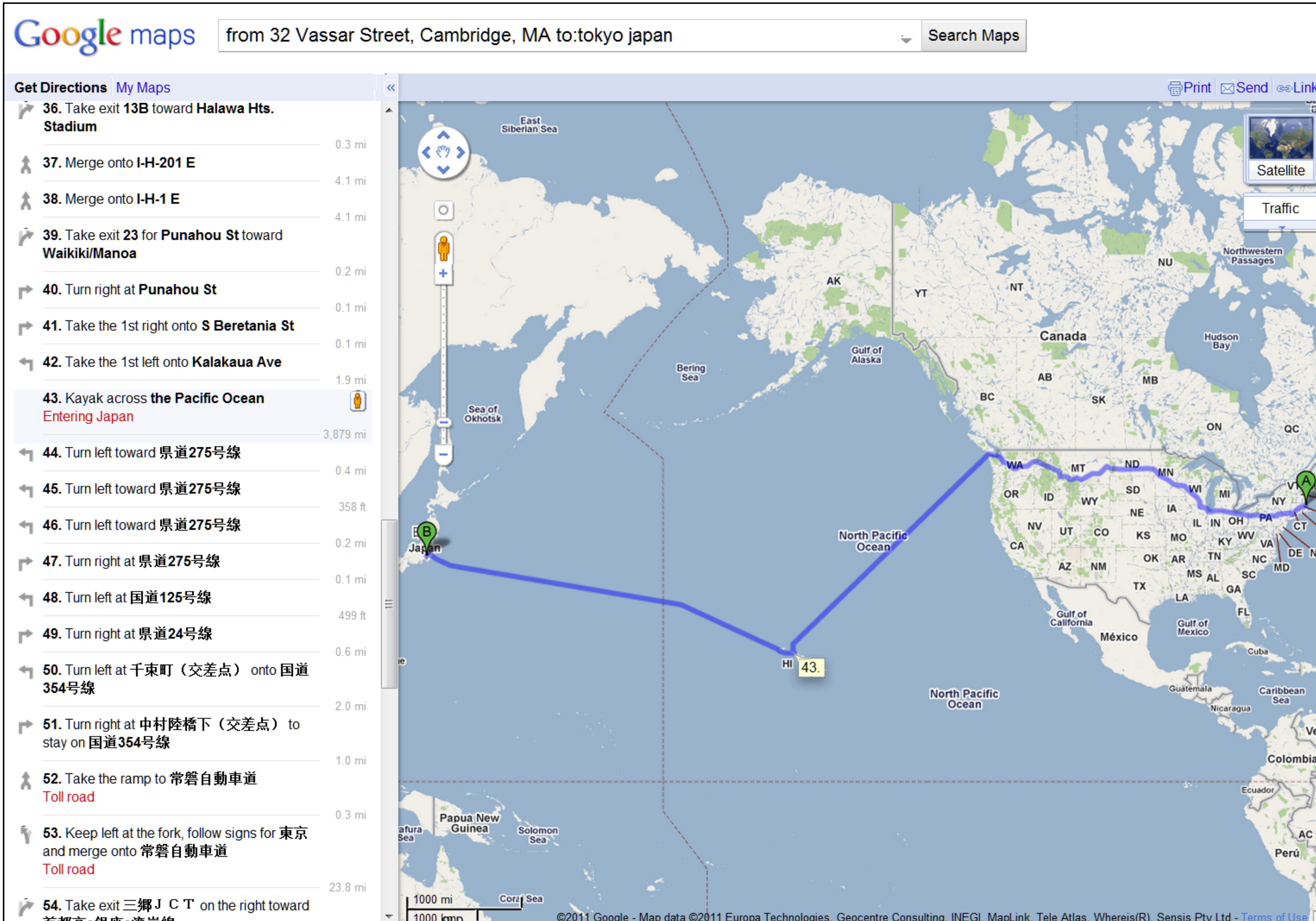
===

Persistent Routes:

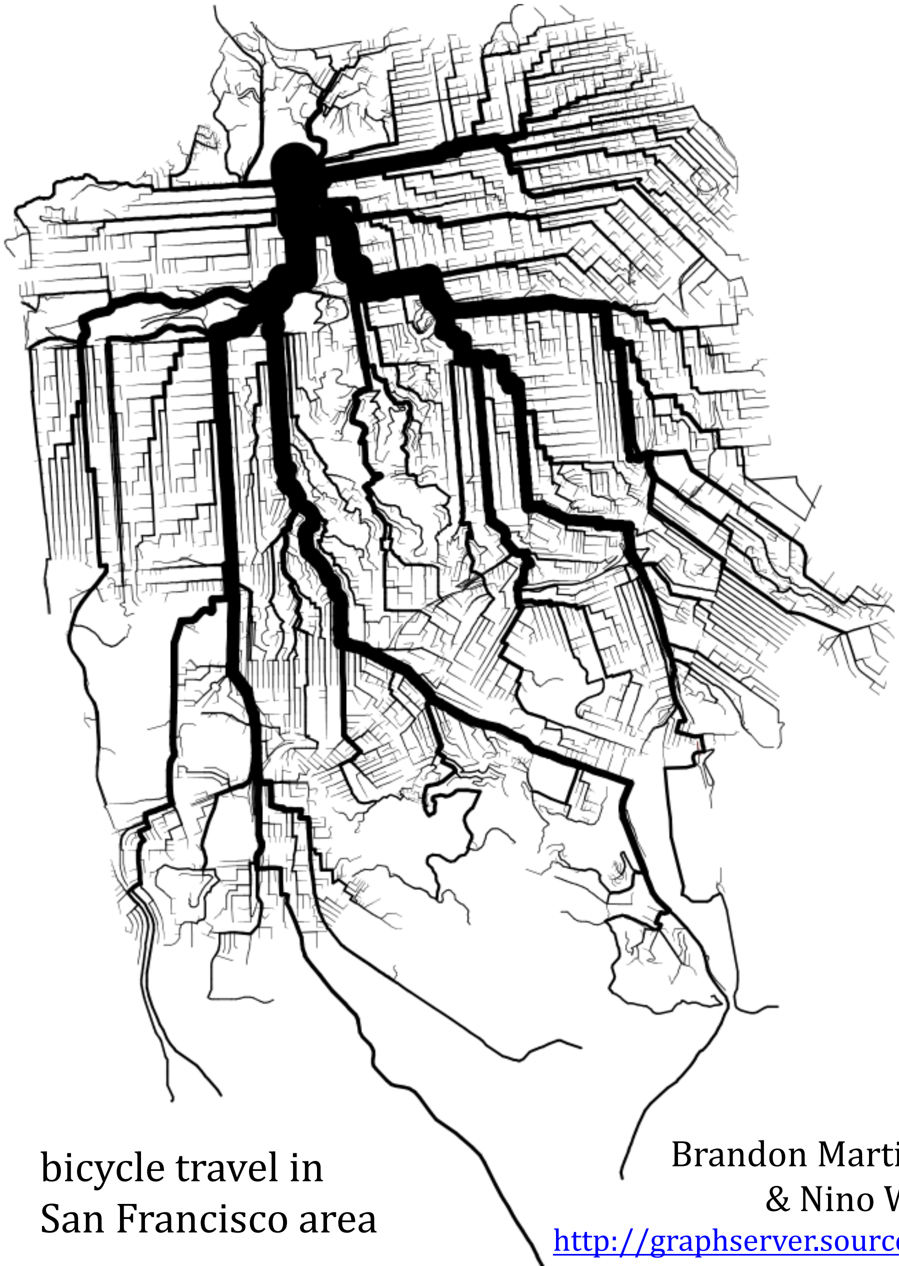
Network Address	Netmask	Gateway Address	Metric
0.0.0.0	0.0.0.0	10.8.0.1	Default

Shortest Paths

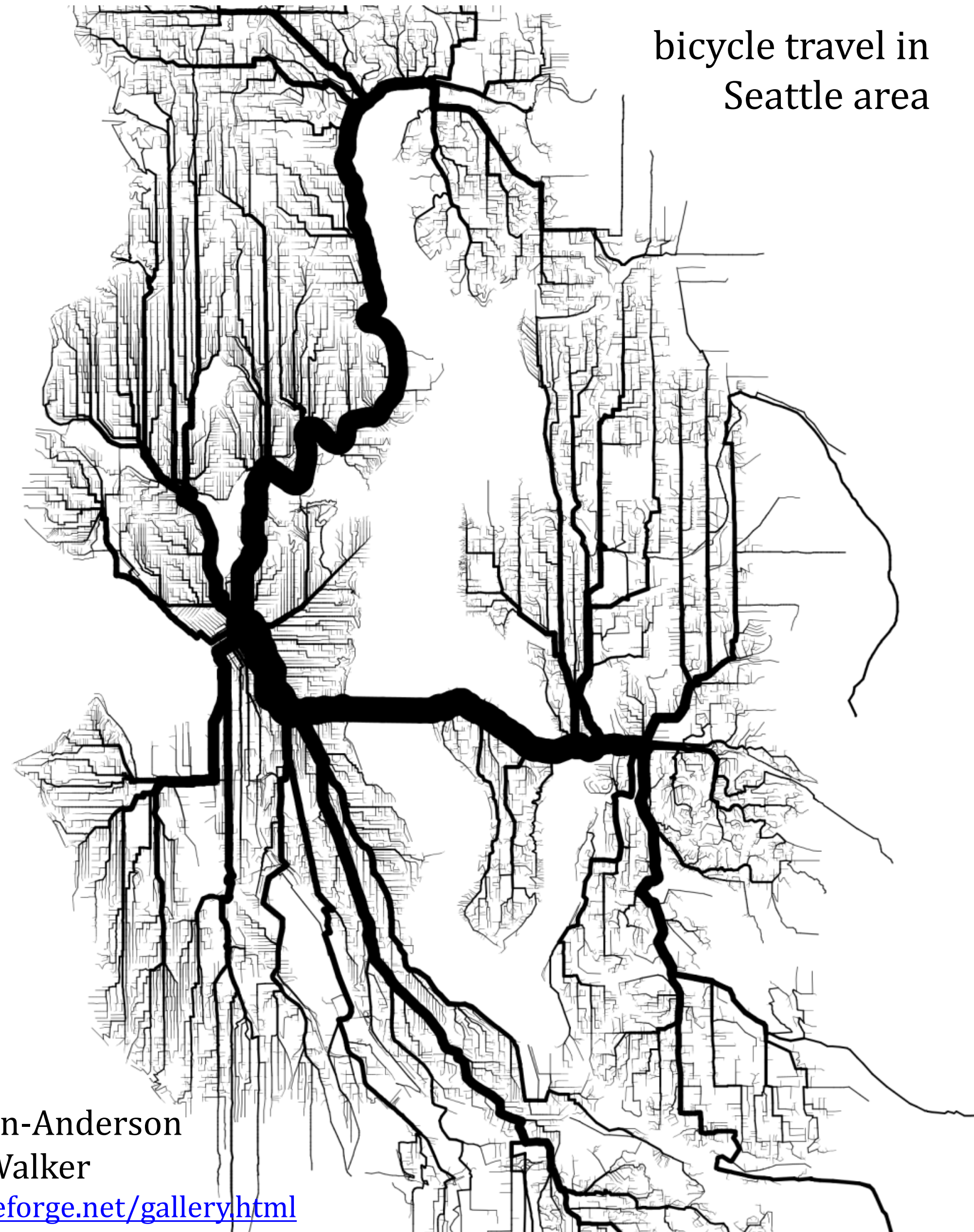
You Can’t Get There From Here



Shortest-Path Trees



bicycle travel in
San Francisco area



bicycle travel in
Seattle area

Brandon Martin-Anderson
& Nino Walker
http://graphserver.sourceforge.net/gallery.html

Relaxation

- In general, refers to letting a solution (temporarily) violate a constraint, and trying to fix these violations

Magic Geek
http://www.youtube.com/watch?v=Y12daEZTUYo