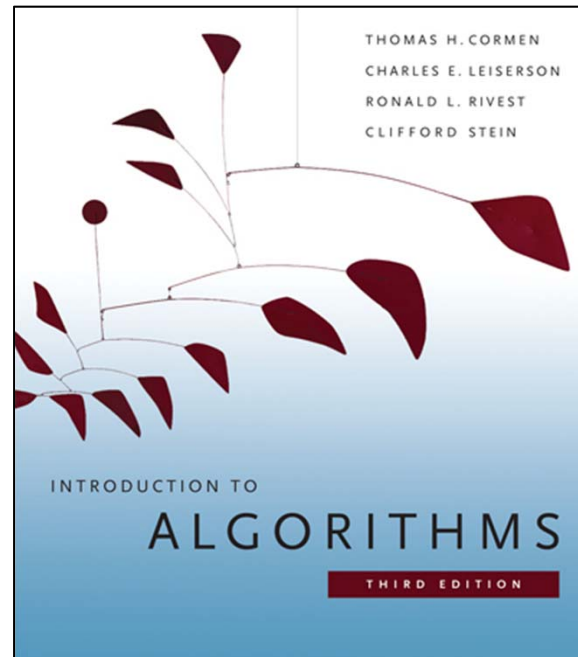


6.006

# *Introduction to Algorithms*



## **Lecture 14: Shortest Paths I**

Prof. Erik Demaine

# Today

- Shortest paths
- Negative-weight cycles
- Triangle inequality
- Relaxation algorithm
- Optimal substructure

# Shortest Paths

Google maps from: 32 Vassar Street, Cambridge, MA to: Times Square, New York, NY Search Maps

Get Directions My Maps

32 Vassar St, Cambridge, Middlesex, Massach  
Times Square, New York, NY  
Add Destination - Show options  
Get Directions

Driving directions to Theater District - Times Square, New York, NY

Suggested routes

Route	Distance	Time
1. I-84 W	221 mi	3 hours 53 mins
2. I-90 W	209 mi	3 hours 58 mins
3. I-395 S and I-95 S	227 mi	4 hours 13 mins

This route has tolls.

32 Vassar St  
Cambridge, MA 02139


- Head southwest on Vassar St
- Turn right at Memorial Dr
- Turn left at Western Ave
- Turn left at Soldiers Field Rd
- Take the I-90 ramp  
Toll road
- Keep right at the fork and merge onto I-90 W  
Partial toll road
- Take exit 9 to merge onto I-84 W toward US-20/Hartford/New York City  
Partial toll road
- Take exit 20 for I-684 toward NY-22/White Plains/Pawling
- Keep left at the fork and merge onto I-684 S
- Merge onto Hutchinson River Pkwy S
- Continue onto Cross County Pkwy (signs for George Washington Bridge)

©2011 Google - Map data ©2011 Google - Terms of Use Report a problem




# Shortest Paths

oracleofbacon.org/cgi-bin/movielinks



## THE ORACLE OF BACON



Welcome  
Credits  
How it Works  
Contact Us  
Other stuff »

© 1999-2010 by Patrick Reynolds. All rights reserved.

Dr. Erik D. Demaine has a Bacon number of 3.

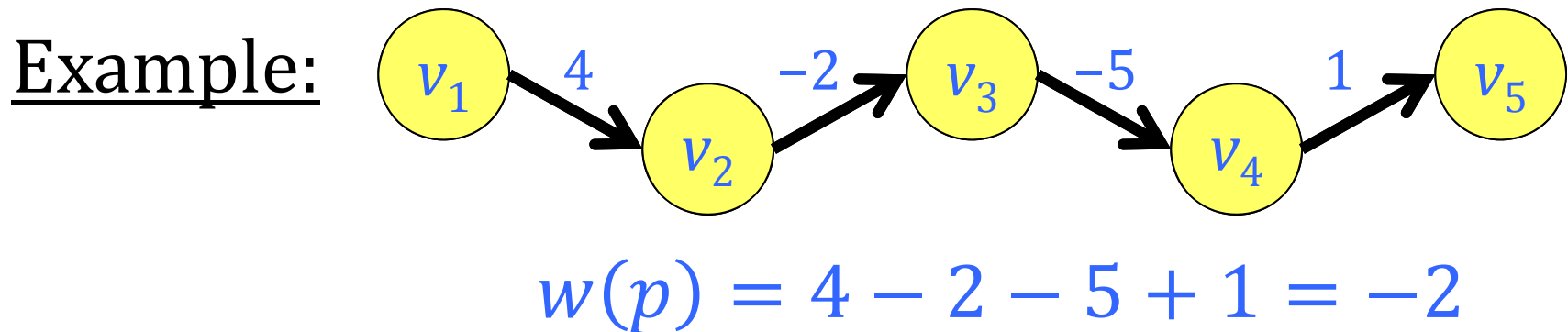
[Find a different link](#)

```
graph TD; A[Dr. Erik D. Demaine] -- was in --> B["The Man Who Saved Geometry (2009) (TV)"]; B -- with --> C[Benoît B. Mandelbrot]; C -- was in --> D["The Revenge of the Dead Indians (1993)"]; D -- with --> E[Dennis Hopper]; E -- was in --> F["The 2004 IFP/West Independent Spirit Awards (2004) (TV)"]; F -- with --> G[Kevin Bacon];
```

Kevin Bacon to Dr. Erik D. Demaine [Find link](#) [More options >>](#)

# How Long Is Your Path?

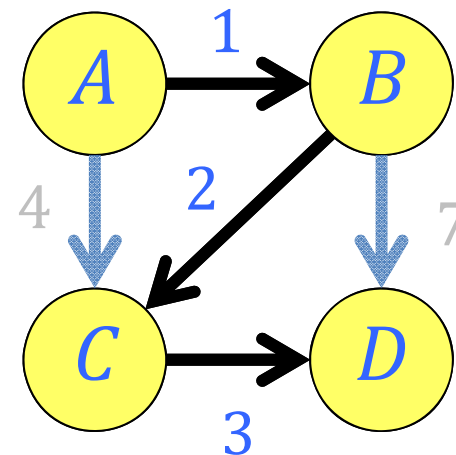
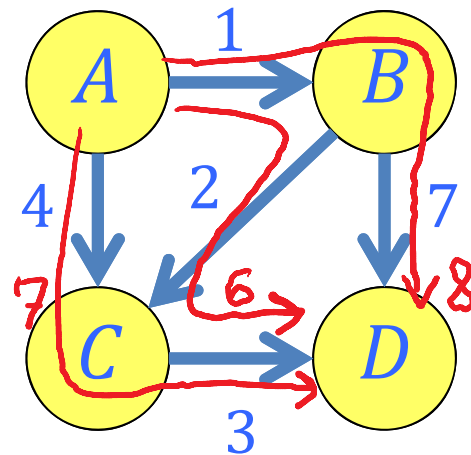
- Directed graph  $G = (V, E)$
- Edge-weight function  $w : E \rightarrow \mathbb{R}$
- Path  $p = v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k$
- **Weight** of  $p$ , denoted  $w(p)$ , is  
 $w(v_1, v_2) + w(v_2, v_3) + \dots + w(v_{k-1}, v_k)$



# My Path Is Shorter Than Yours

- A *shortest path* from  $u$  to  $v$  is a path  $p$  of minimum possible weight  $w(p)$  from  $u$  to  $v$
- The shortest-path weight  $\delta(u, v)$  from  $u$  to  $v$  is the weight of any such shortest path:  
$$\delta(u, v) = \min \{w(p) : p \text{ is a path from } u \text{ to } v\}$$

Example:



$$\delta(A, D) = 6$$

# You Can't Get There From Here

Google maps from 32 Vassar Street, Cambridge, MA to:tokyo japan Search Maps

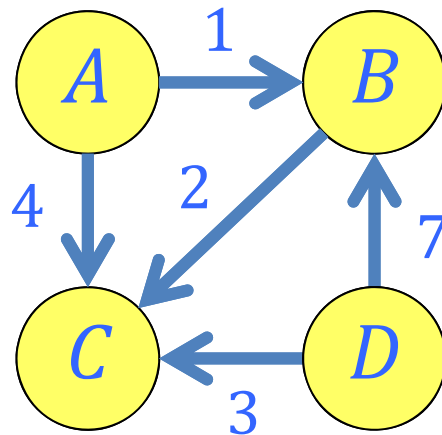
Get Directions My Maps

- 36. Take exit 13B toward Halawa Hts. Stadium 0.3 mi
- 37. Merge onto I-H-201 E 4.1 mi
- 38. Merge onto I-H-1 E 4.1 mi
- 39. Take exit 23 for Punahou St toward Waikiki/Manoa 0.2 mi
- 40. Turn right at Punahou St 0.1 mi
- 41. Take the 1st right onto S Beretania St 0.1 mi
- 42. Take the 1st left onto Kalakaua Ave 1.9 mi
- 43. Kayak across the Pacific Ocean Entering Japan** 3,879 mi
- 44. Turn left toward 県道275号線 0.4 mi
- 45. Turn left toward 県道275号線 358 ft
- 46. Turn left toward 県道275号線 0.2 mi
- 47. Turn right at 県道275号線 0.1 mi
- 48. Turn left at 国道125号線 499 ft
- 49. Turn right at 県道24号線 0.6 mi
- 50. Turn left at 千束町 (交差点) onto 国道354号線 2.0 mi
- 51. Turn right at 中村陸橋下 (交差点) to stay on 国道354号線 1.0 mi
- 52. Take the ramp to 常磐自動車道 Toll road 0.3 mi
- 53. Keep left at the fork, follow signs for 東京 and merge onto 常磐自動車道 Toll road 23.8 mi
- 54. Take exit 三郷 J C T on the right toward 首都高・銀座・芝浦線

# You Can't Get There From Here

- If there is no path from  $u$  to  $v$ , then neither is there a *shortest* path from  $u$  to  $v$
- Define  $\delta(u, v) = \infty$  in this case

Example:



$$\delta(A, D) = \infty$$



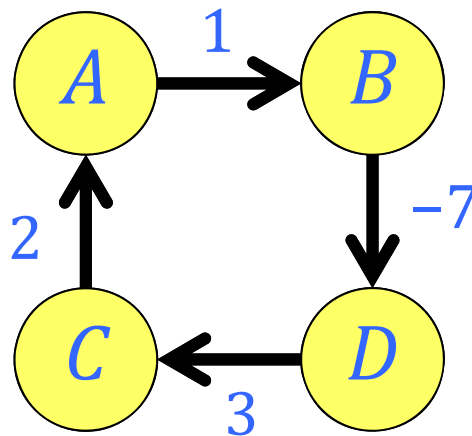
# The More I Walk, The Less It Takes

- A shortest path from  $u$  to  $v$  might not exist, even though there is a path from  $u$  to  $v$
- ***Negative-weight cycle***

$$c = v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k \rightarrow v_1$$

has  $w(c) < 0$

Example:

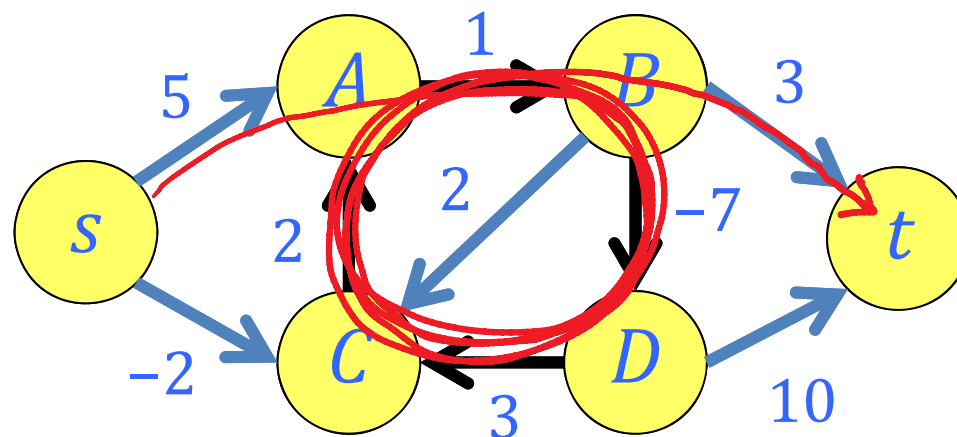


$$\begin{aligned} w(A \rightarrow B \rightarrow C \\ \rightarrow D \rightarrow A) \\ = -1 \end{aligned}$$

# The More I Walk, The Less It Takes

- Define  $\delta(u, v) = -\infty$  if there's a path from  $u$  to  $v$  that visits a negative-weight cycle
- $\delta(u, v) = \inf \{w(p) : p \text{ is a path from } u \text{ to } v\}$

Example:

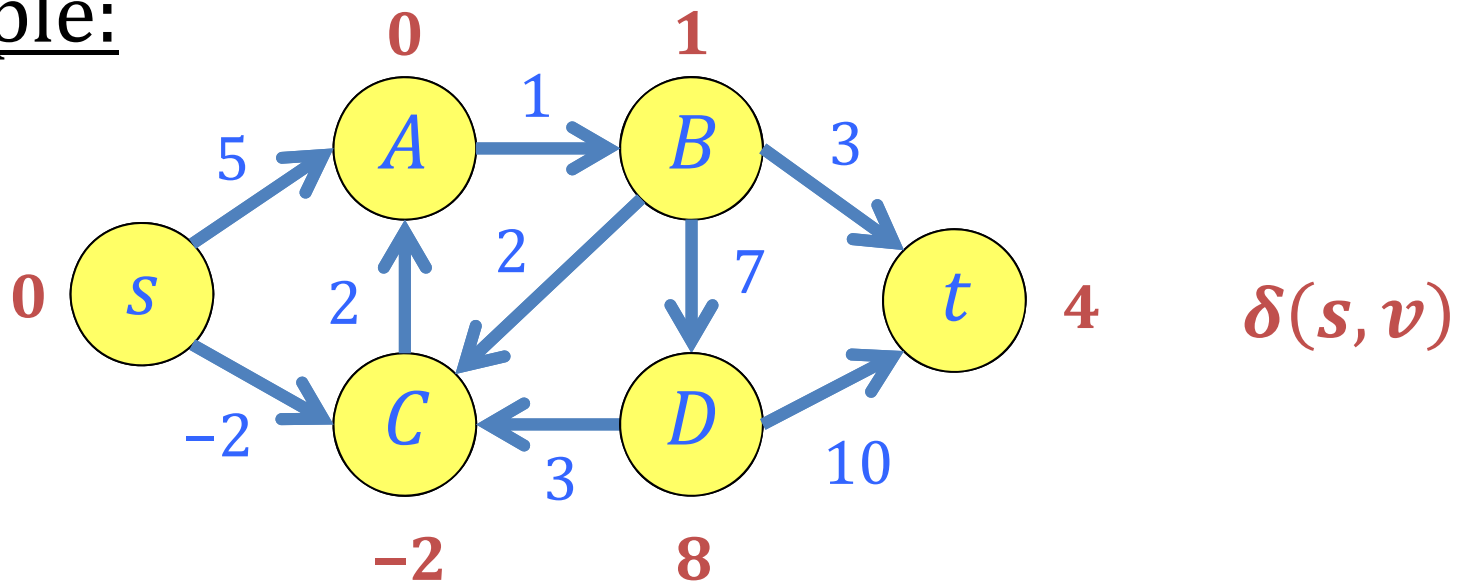


$$\delta(s, t) = -\infty$$

# Single-Source Shortest Paths

- Problem: Given a directed graph  $G = (V, E)$  with edge-weight function  $w : E \rightarrow \mathbb{R}$ , and a **source** vertex  $s$ , compute  $\delta(s, v)$  for all  $v \in V$

Example:

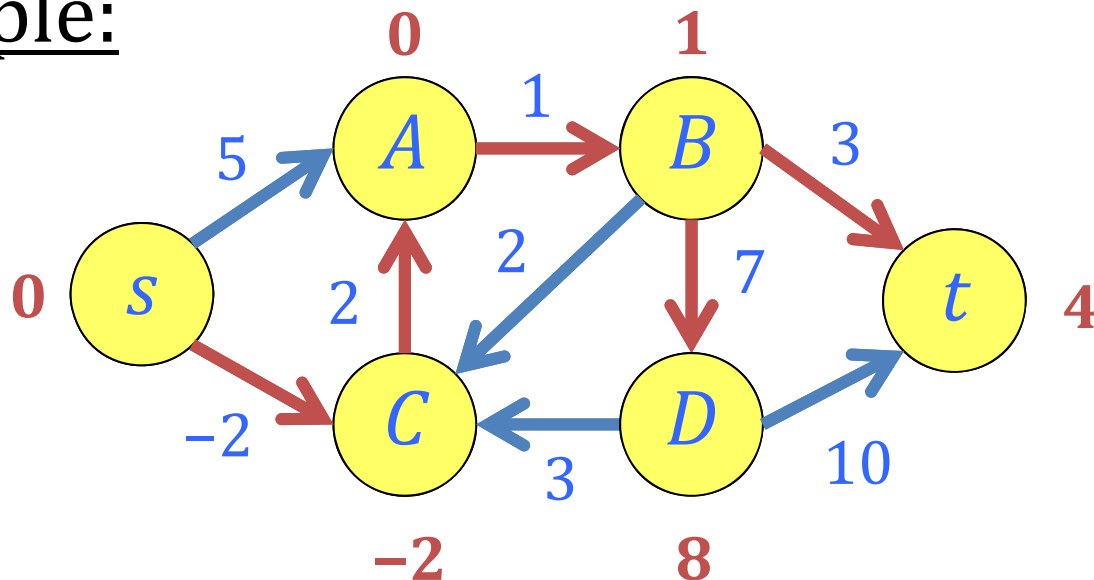


# Shortest-Path Tree

- Ideally also compute a *shortest-path tree* containing a shortest path from source  $s$  to every  $v \in V$  (assuming shortest paths exist)
  - Represent by storing *parent*  $v.\pi$  for each  $v \in V$

*=  $\pi[v]$  in earlier CLR(s)*

Example:

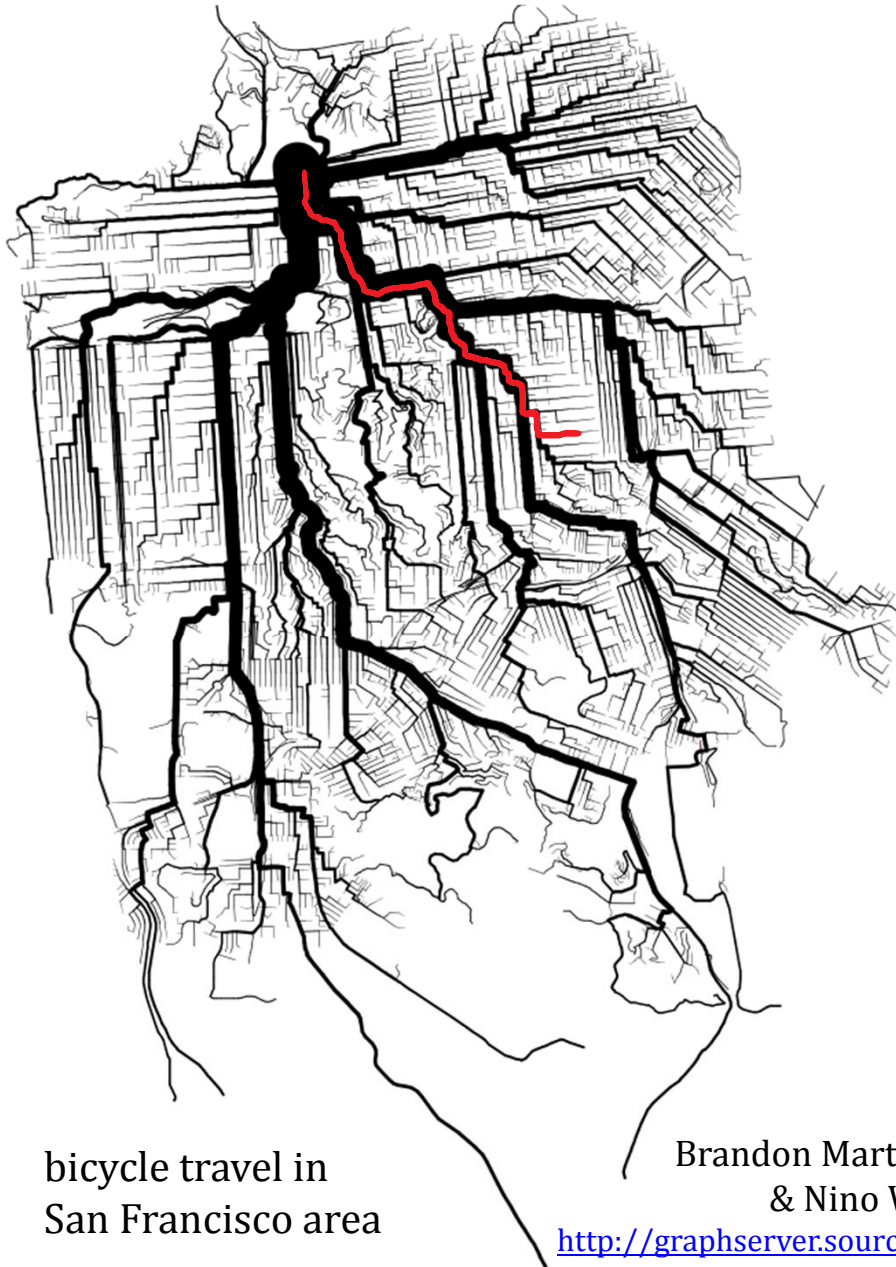


*t.π = B  
B.π = A  
D.π = B*

*δ(s, v)*

*A.π = C  
C.π = s  
s.π = None*

# 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>



# Single-Source Shortest-Path Algorithms

- **Relaxation algorithm** *(TODAY)*
  - Framework for most shortest-path algorithms
  - Not necessarily efficient
- **Bellman-Ford algorithm** *(LECTURE 15)*
  - Deals with negative weights
  - Slow but polynomial
- **Dijkstra's algorithm** *(LECTURE 16)*
  - Fast (nearly linear time)
  - Requires nonnegative weights

# Brute-Force Algorithm

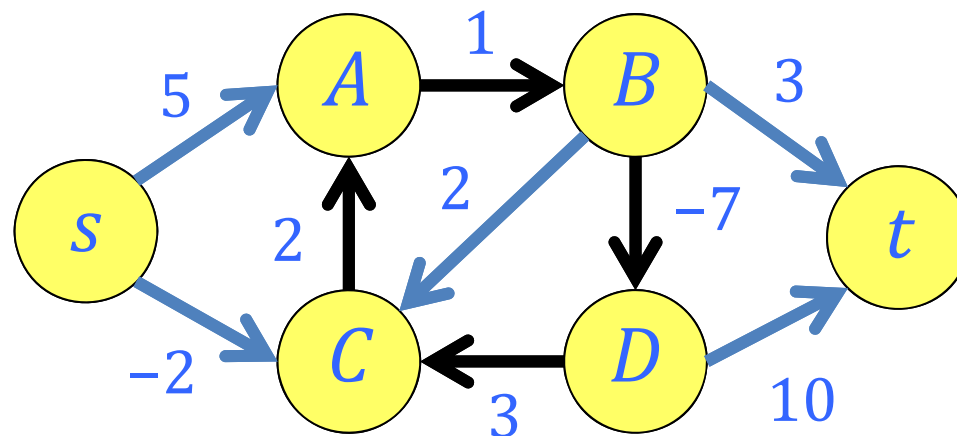
distance( $s, t$ ):

for each path  $p$  from  $s$  to  $t$ :

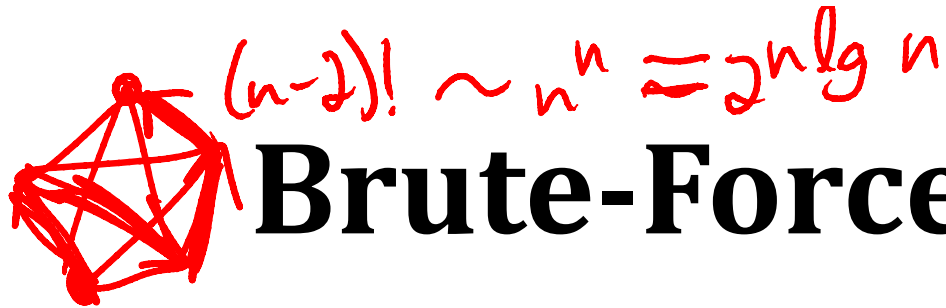
compute  $w(p)$

return  $p$  encountered with smallest  $w(p)$

- Number of paths can be *infinite*:



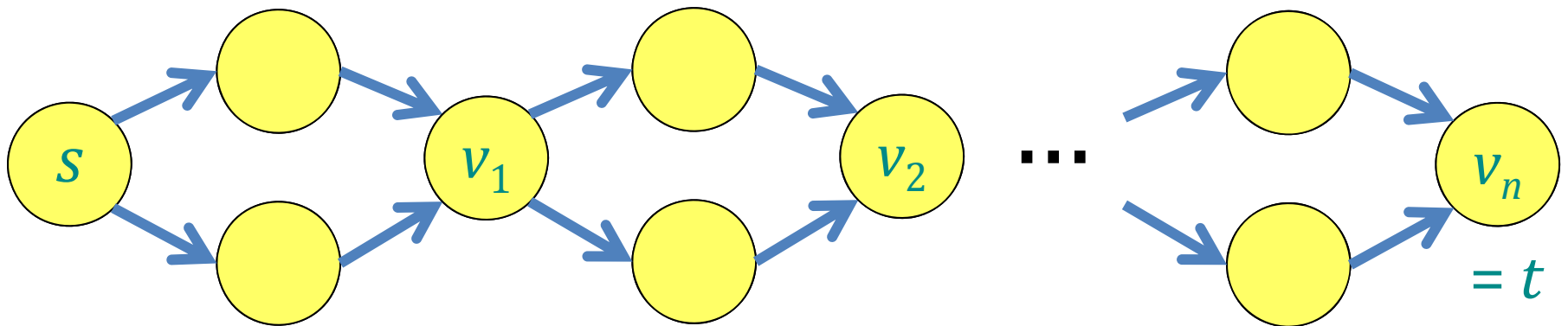
$$\delta(s, t) = -\infty$$



# Brute-Force Algorithm

distance( $s, t$ ): *## assume no negative-weight cycles*  
for each simple path  $p$  from  $s$  to  $t$ :  $\leftarrow$  # paths  
    compute  $w(p)$   $\leftarrow O(V)$   
return  $p$  encountered with smallest  $w(p)$

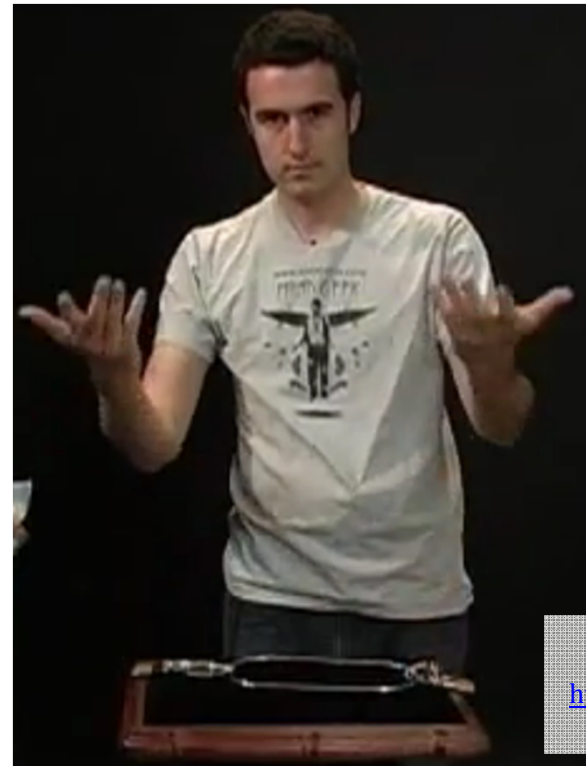
- Number of paths can be *exponential*:



$2^n$  paths from  $s$  to  $v_n$ ;  $O(n)$  vertices and edges

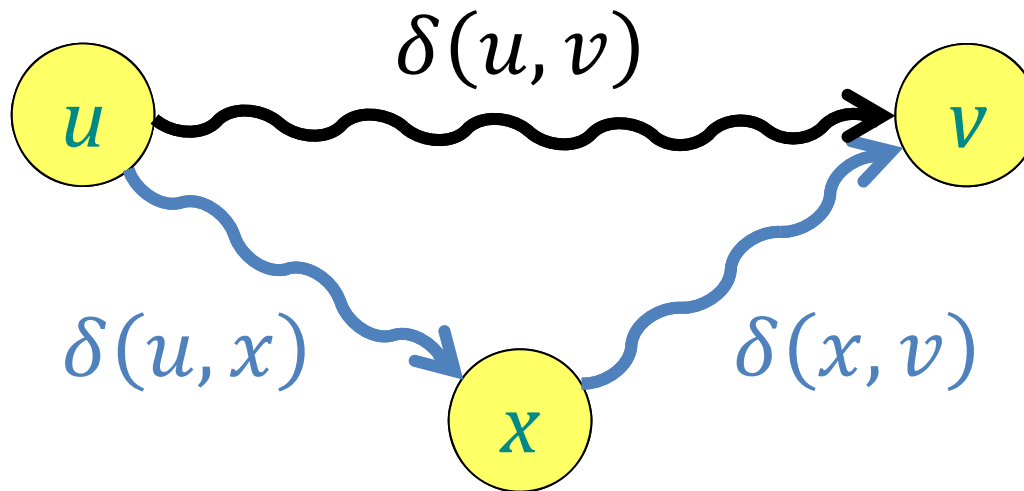
# Relaxation

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



# Triangle Inequality

- Theorem: For all  $u, v, x \in V$ , we have  
$$\delta(u, v) \leq \delta(u, x) + \delta(x, v).$$



- Proof: Shortest path from  $u$  to  $v$  is at most any particular path, e.g., the blue chain. ■



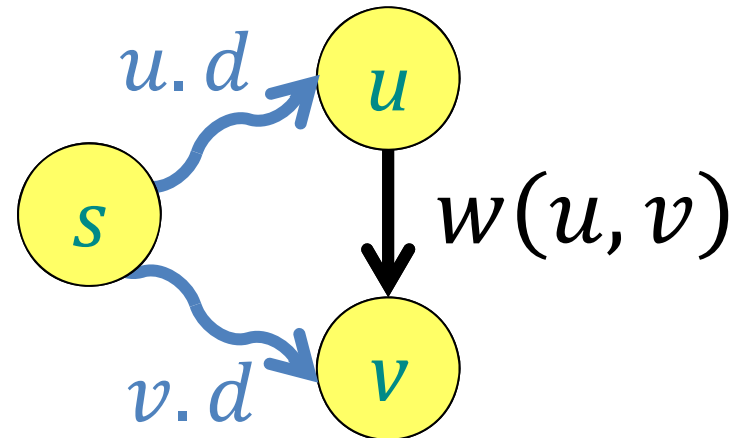
# Relaxation Approach

- Maintain *distance estimate*  $v.d = d[v]$  in older CLR(s) for each  $v \in V$
- Goal:  $v.d = \delta(s, v)$  for all  $v \in V$
- Invariant:  $v.d \geq \delta(s, v)$
- Initialization:

for $v$ in $V$ :
$v.d = \infty$
$s.d = 0$
- Repeatedly improve estimates toward goal, by aiming to achieve triangle inequality

# Edge Relaxation

- Consider an edge  $(u, v)$



- $\delta(s, v) \leq \delta(s, u) + \delta(u, v)$   
 $\leq \delta(s, u) + w(u, v)$

[triangle ineq.]  
[candidate path]

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

**relax** $(u, v)$ :

if  $v.d > u.d + w(u, v)$ :

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

# Relaxation Algorithm

for  $v$  in  $V$ :

$$v.d = \infty$$

$$s.d = 0$$

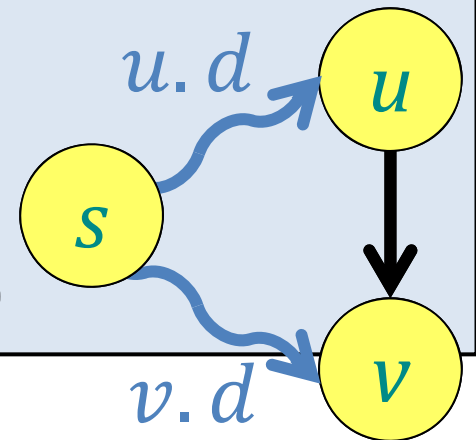
while some edge  $(u, v)$  has  $v.d > u.d + w(u, v)$ :

pick such an edge  $(u, v)$

**relax** $(u, v)$ :

if  $v.d > u.d + w(u, v)$ :

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



# Relaxation Algorithm with Shortest-Path Tree

for  $v$  in  $V$ :

$$v.d = \infty$$

$$v.\pi = \text{None}$$

$$s.d = 0$$

while some edge  $(u, v)$  has  $v.d > u.d + w(u, v)$ :

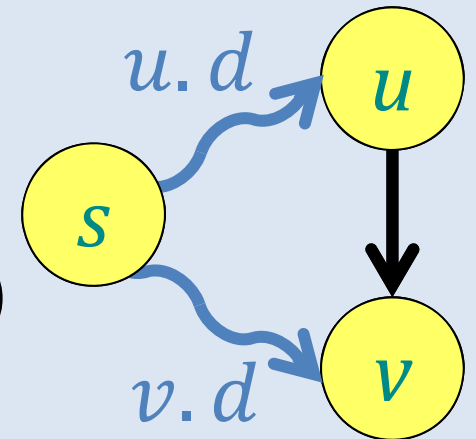
pick such an edge  $(u, v)$

**relax** $(u, v)$ :

if  $v.d > u.d + w(u, v)$ :

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

$$v.\pi = u$$



# Relaxing Is Safe

- Lemma: The relaxation algorithm maintains the invariant that  $v.d \geq \delta(s, v)$  for all  $v \in V$ .
- Proof: By induction on the number of steps.

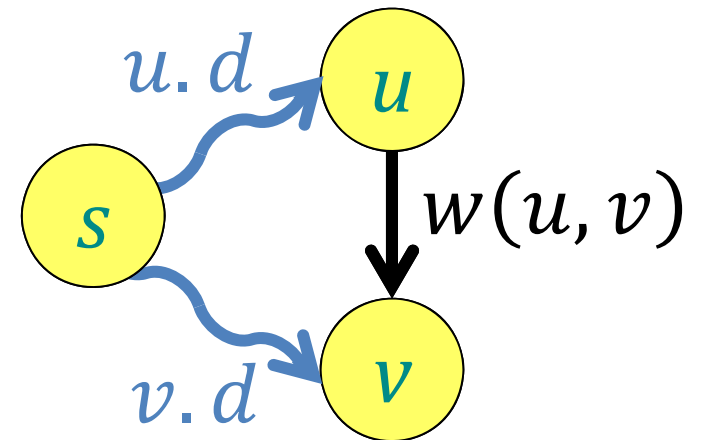
- Consider  $\text{relax}(u, v)$

- By induction,  $u.d \geq \delta(s, u)$

- By triangle inequality,

$$\begin{aligned}\delta(s, v) &\leq \delta(s, u) + \delta(u, v) \\ &\leq u.d + w(u, v)\end{aligned}$$

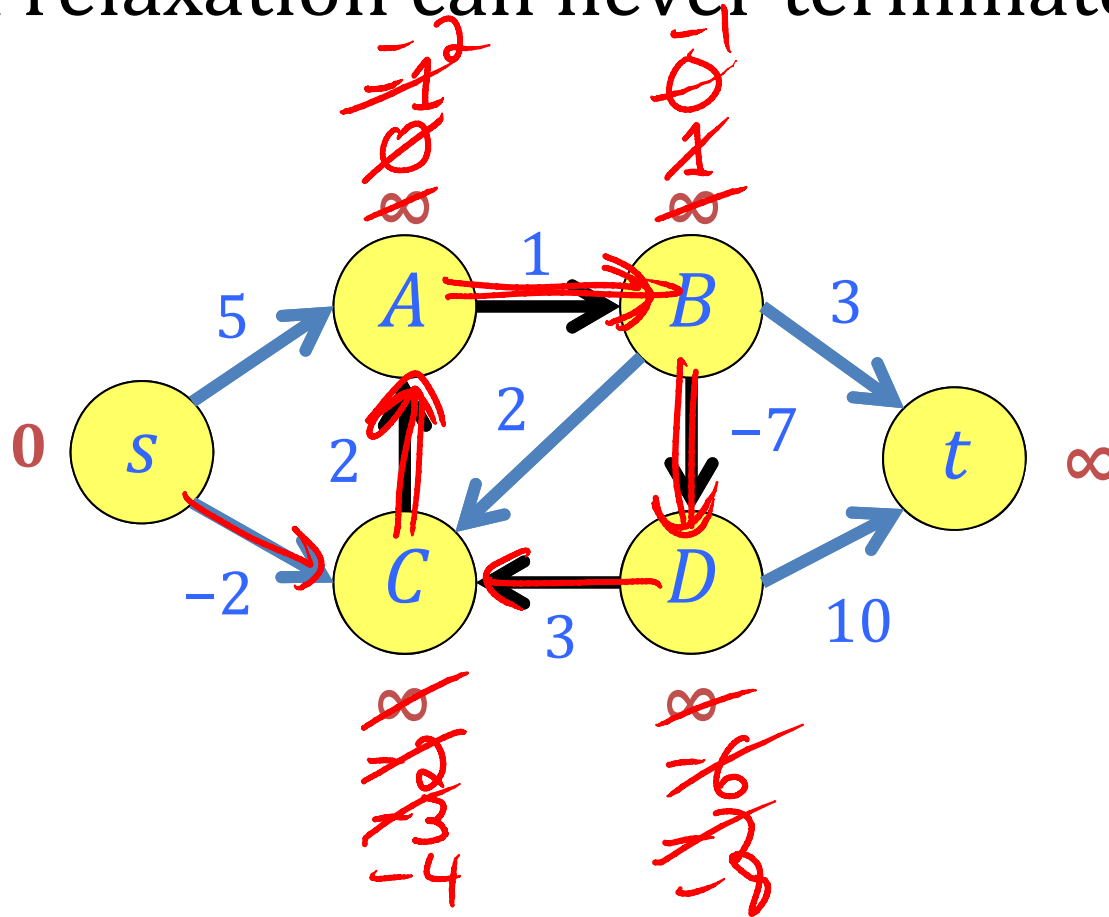
- So setting  $v.d = u.d + w(u, v)$  is “safe” ■



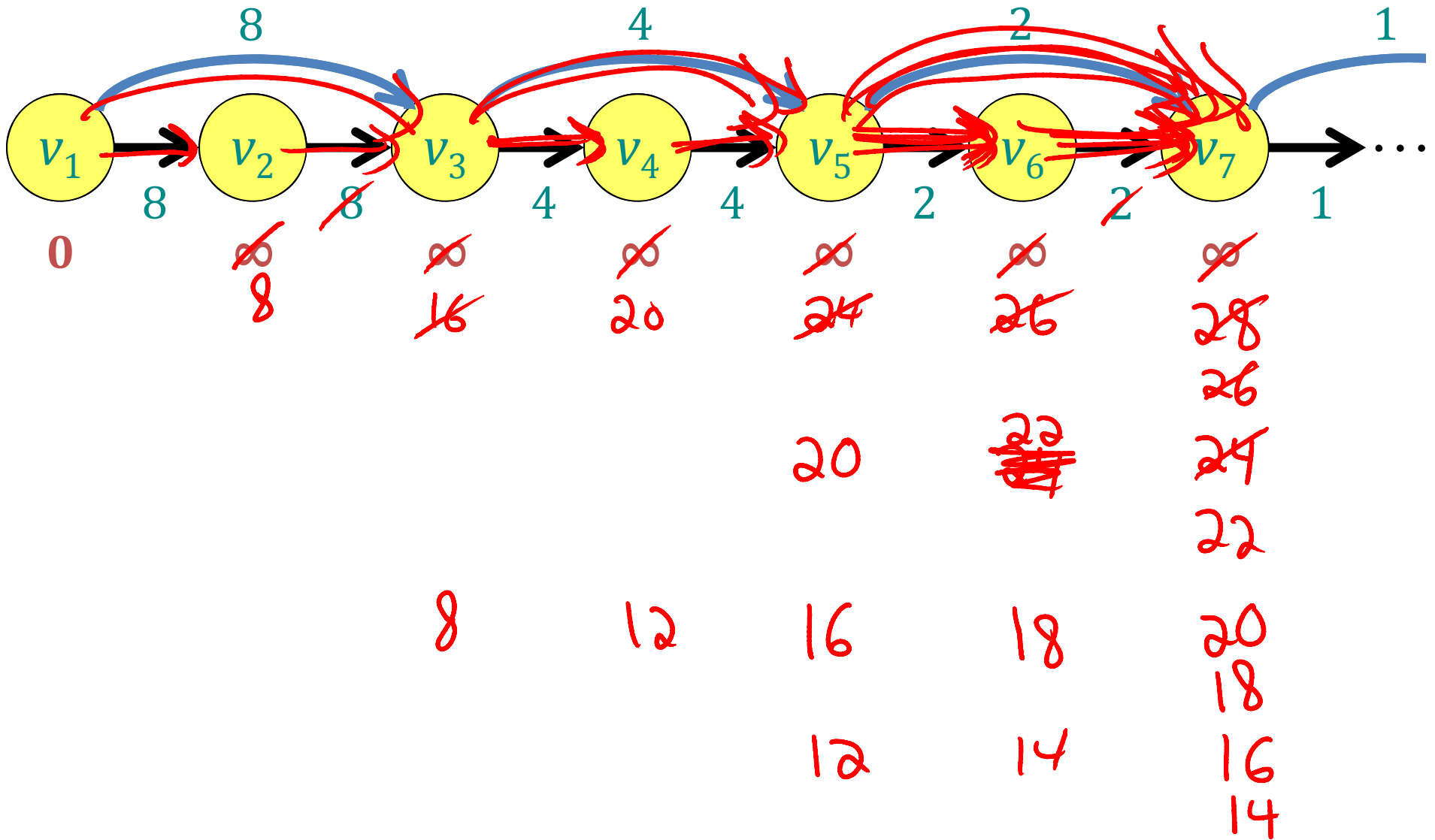


# Infinite Relaxation

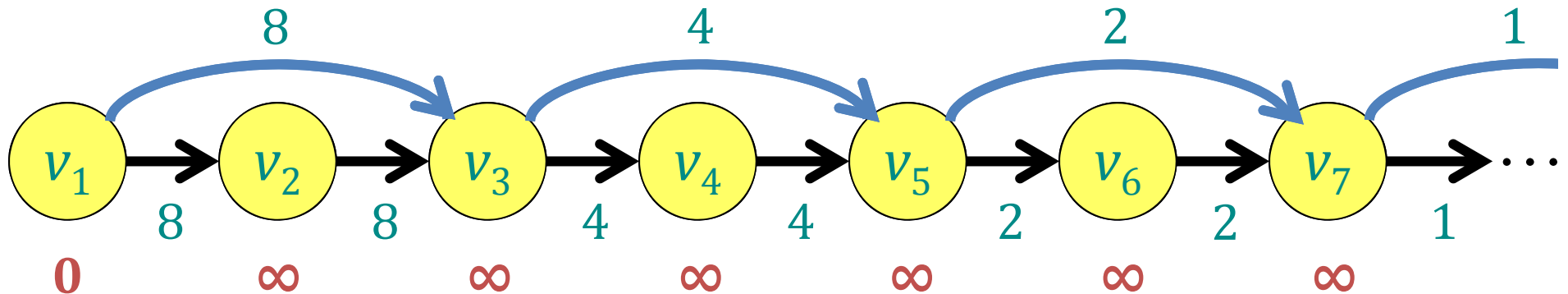
- If a negative-weight cycle is reachable from  $s$ , then relaxation can never terminate



# Long Relaxation



# Long Relaxation



- Analysis:

- relax( $v_1, v_2$ ) - 1

$$T(n) = 2T(n-2) + 3$$

- relax( $v_2, v_3$ ) - 2

$$T(n) = \Theta(2^{n/2})$$

- recurse on  $v_3, v_4, \dots, v_n$  }  $T(n-2)$

- relax( $v_1, v_3$ ) - 3

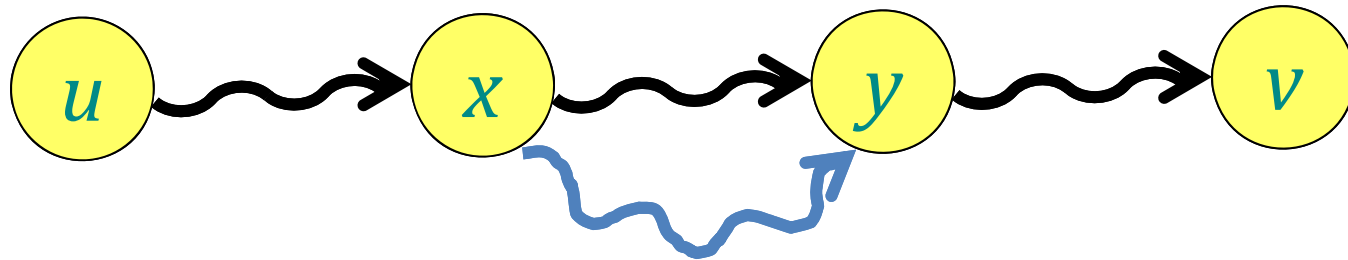
- recurse on  $v_3, v_4, \dots, v_n$  }  $T(n-2)$

# Are You Sure This Is a Good Idea?

- **Bellman-Ford algorithm:** *(LECTURE 15)*
  - Relax all of the edges
  - Repeat  $\sim |V|$  times
  - Polynomial time!
- **Dijkstra's algorithm:** *(LECTURE 16)*
  - Relax edges in a growing ball around  $s$
  - Nearly linear time!
  - (but doesn't work with negative edge weights)

# Optimal Substructure

- Lemma: A subpath of a shortest path is a shortest path (between its endpoints).



- Proof: By contradiction.
  - If there were a shorter path from  $x$  to  $y$ , then we could **shortcut** the path from  $u$  to  $v$ , contradicting that we had a shortest path. ■