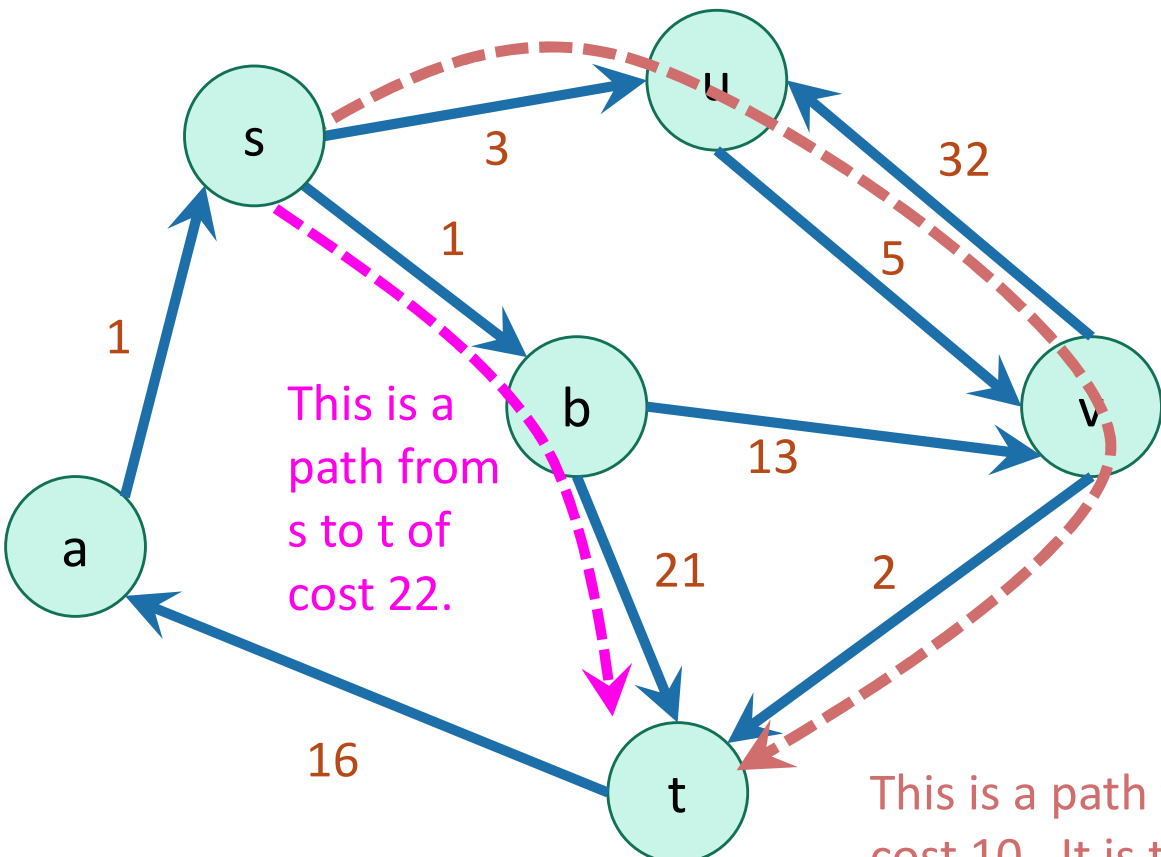
Đồ thị: Đường đi ngắn nhất

Bellman-Ford algorithm

Sử dụng một phần tài liệu bài giảng CS161 Stanford University

Recall

- A weighted directed graph:



This is a path from s to t of cost 10. It is the shortest path from s to t.

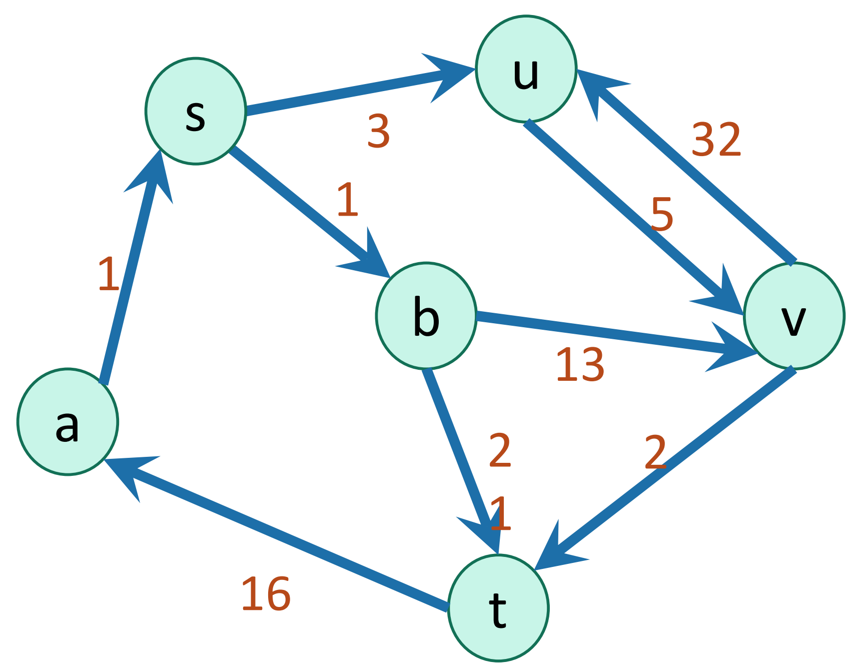
- Weights on edges represent costs.
- The cost of a path is the sum of the weights along that path.
- A shortest path from s to t is a directed path from s to t with the smallest cost.
- The single-source shortest path problem is to find the shortest path from s to v for all v in the graph.

Đã học

- Thuật toán Dijkstra
  - Tìm đường đi ngắn nhất từ một nguồn cho đồ thị có trọng số.

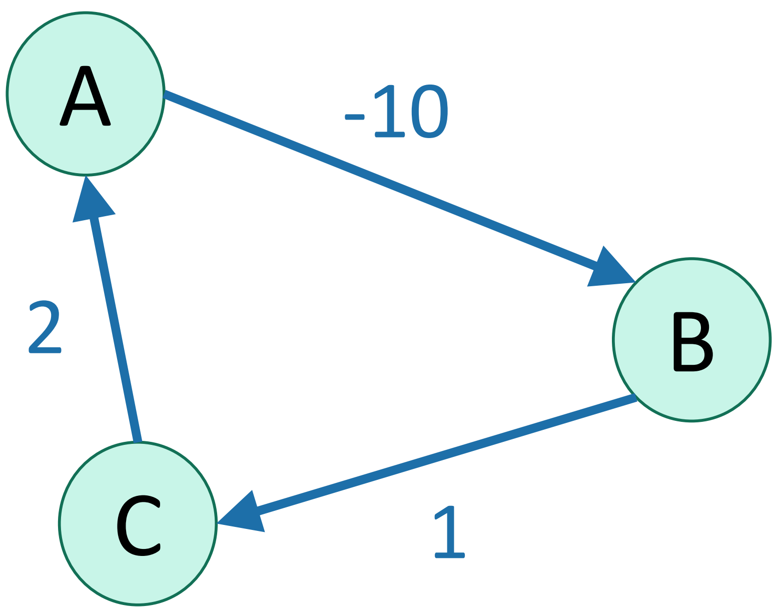
Hạn chế của tt Dijkstra

- Needs non-negative edge weights.
- If the weights change, we need to re-run the whole thing.



Khái niệm chu trình âm

- A **negative cycle** is a cycle whose edge weights sum to a negative number.
- Shortest paths aren't defined when there are negative cycles!

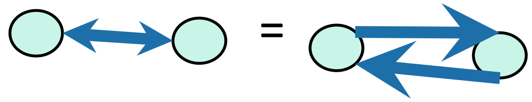


The shortest path from A to B has cost…negative infinity?

Bellman-Ford algorithm

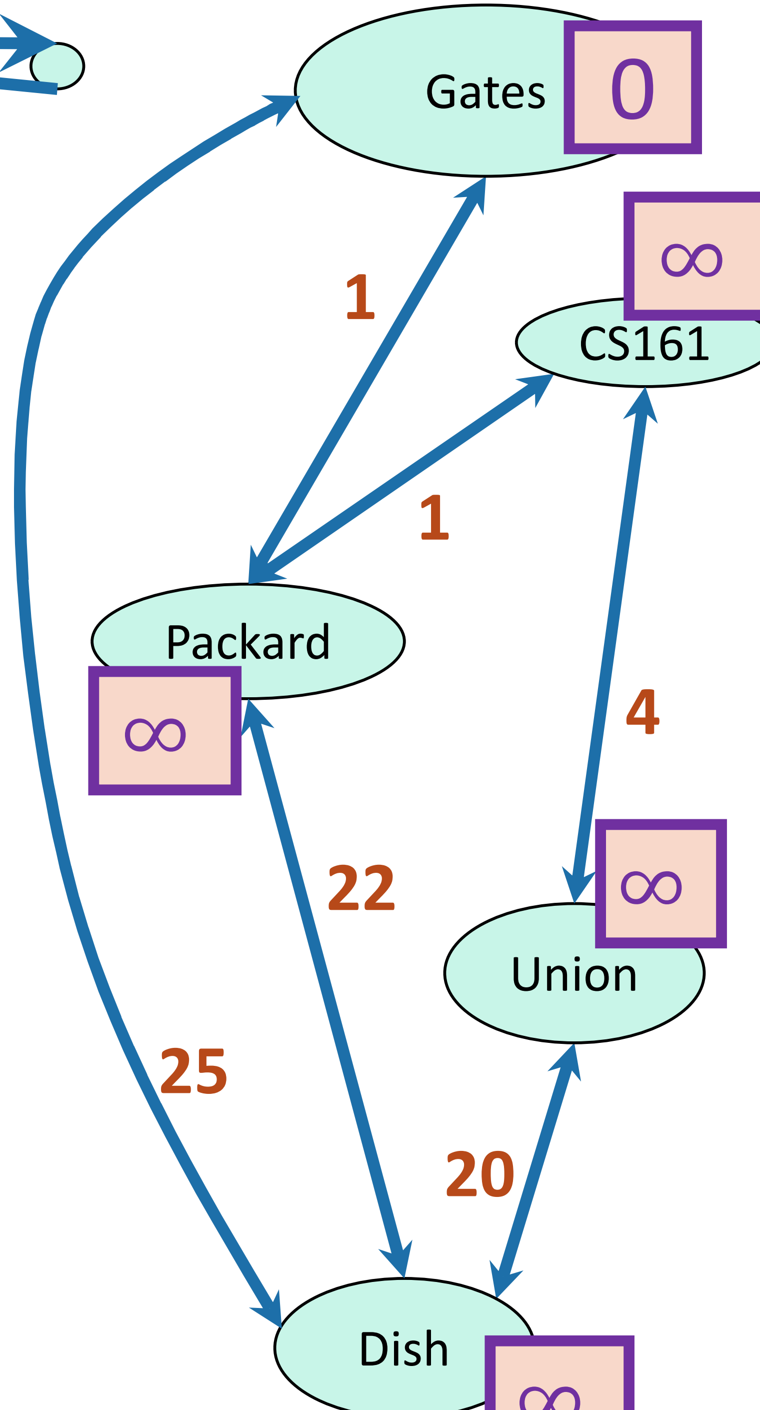
- (-) Slower than Dijkstra’s algorithm
- (+) Can handle negative edge weights
  - Can **detect** negative cycles!

Bellman-Ford



How far is a node from Gates?

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$					
$d^{(2)}$					
$d^{(3)}$					
$d^{(4)}$					

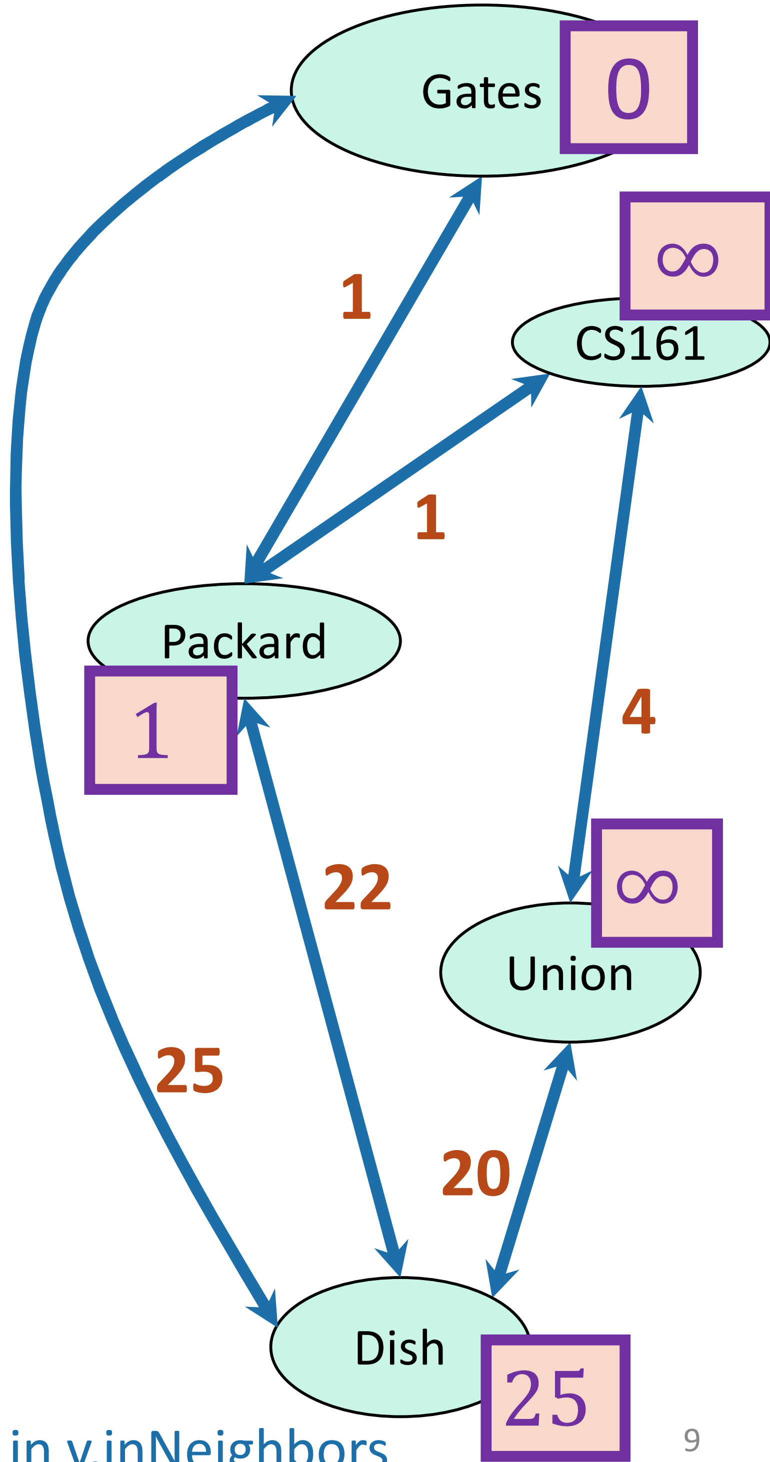


- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , d^{(i)}[u] + w(u,v))$
      where we are also taking the min over all u in v.inNeighbors

Bellman-Ford

How far is a node from Gates?

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$					
$d^{(3)}$					
$d^{(4)}$					

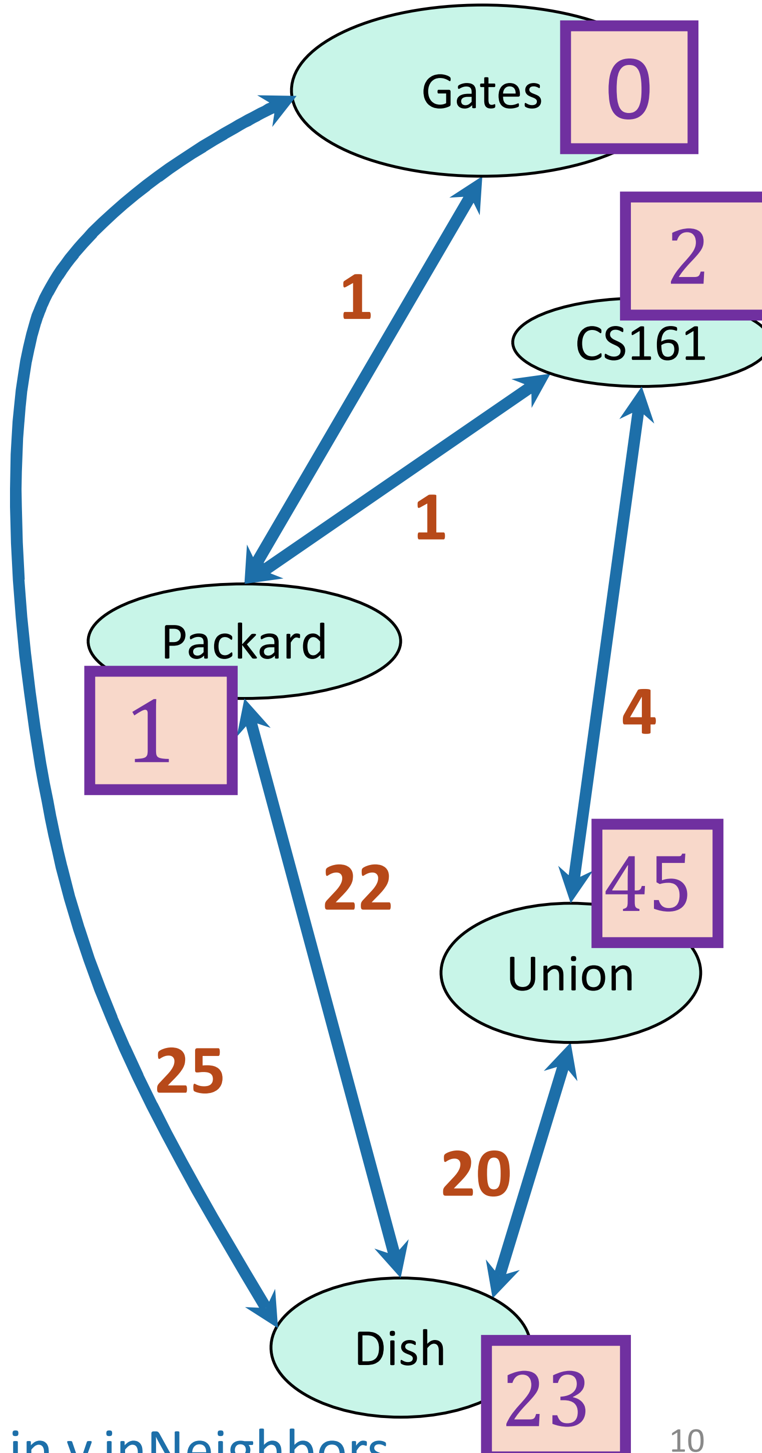


- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , d^{(i)}[u] + w(u,v))$
      where we are also taking the min over all u in v.inNeighbors

Bellman-Ford

How far is a node from Gates?

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$					
$d^{(4)}$					

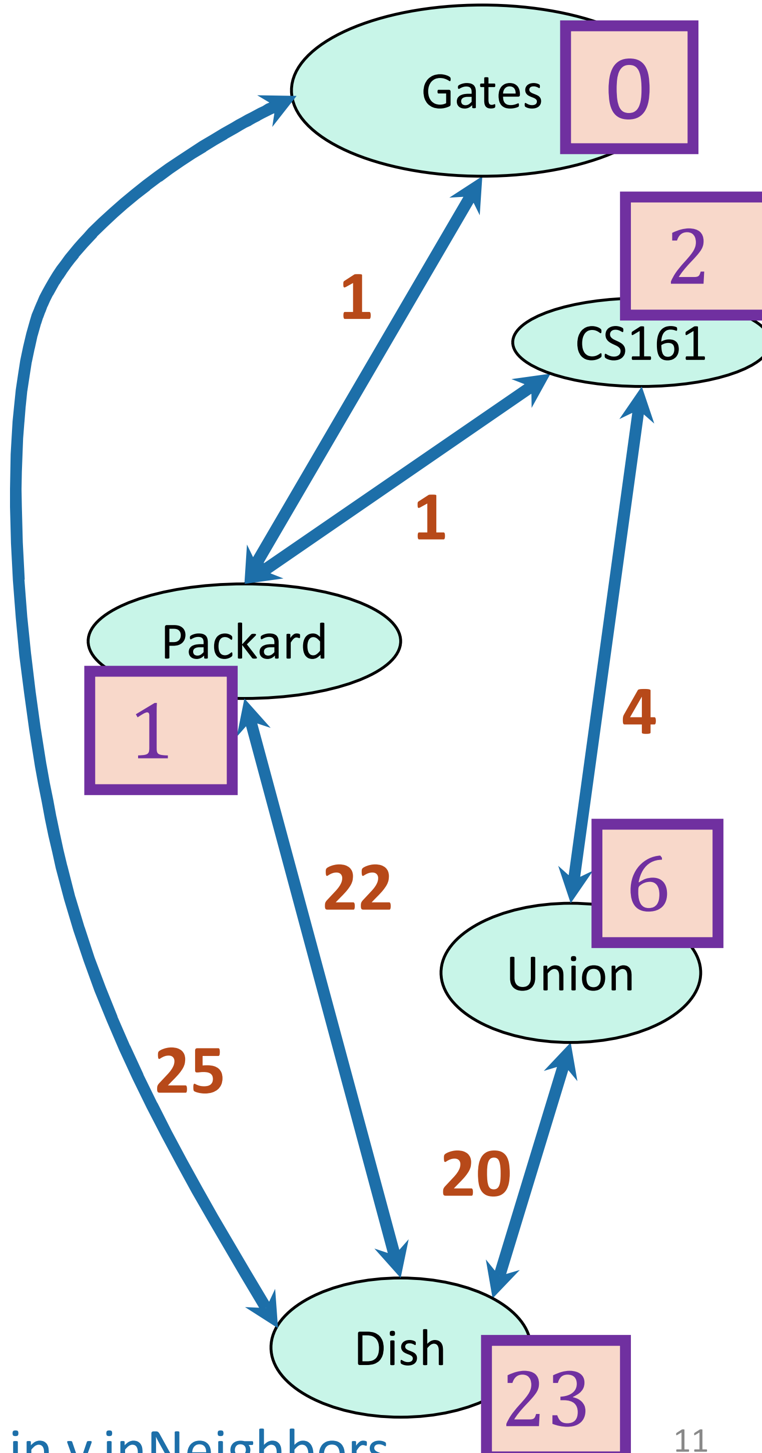


- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , d^{(i)}[u] + w(u,v))$
      where we are also taking the min over all u in v.inNeighbors

Bellman-Ford

How far is a node from Gates?

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$	0	1	2	**6**	23
$d^{(4)}$					



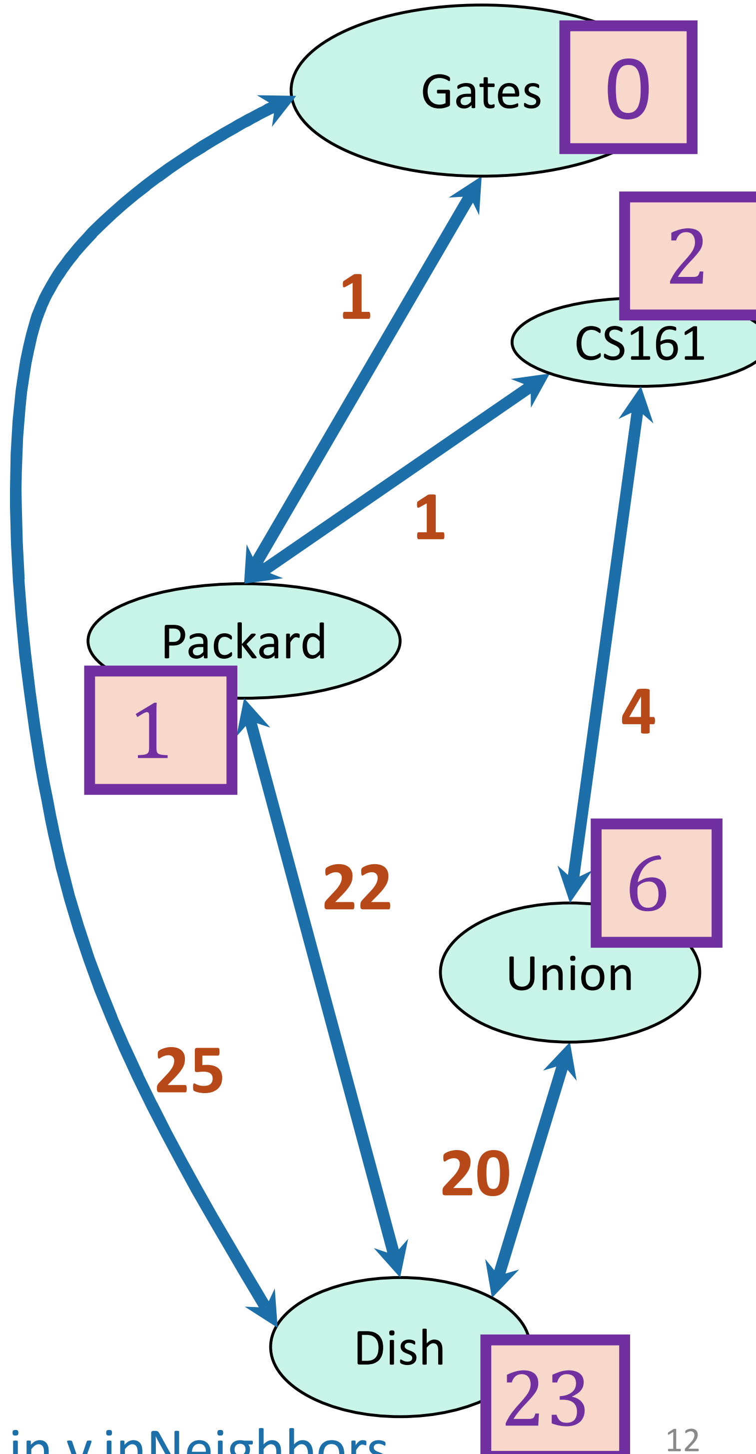
- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , d^{(i)}[u] + w(u,v))$
      where we are also taking the min over all u in v.inNeighbors

Bellman-Ford

How far is a node from Gates?

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$	0	1	2	**6**	23
$d^{(4)}$	0	1	2	6	23

These are the final distances!

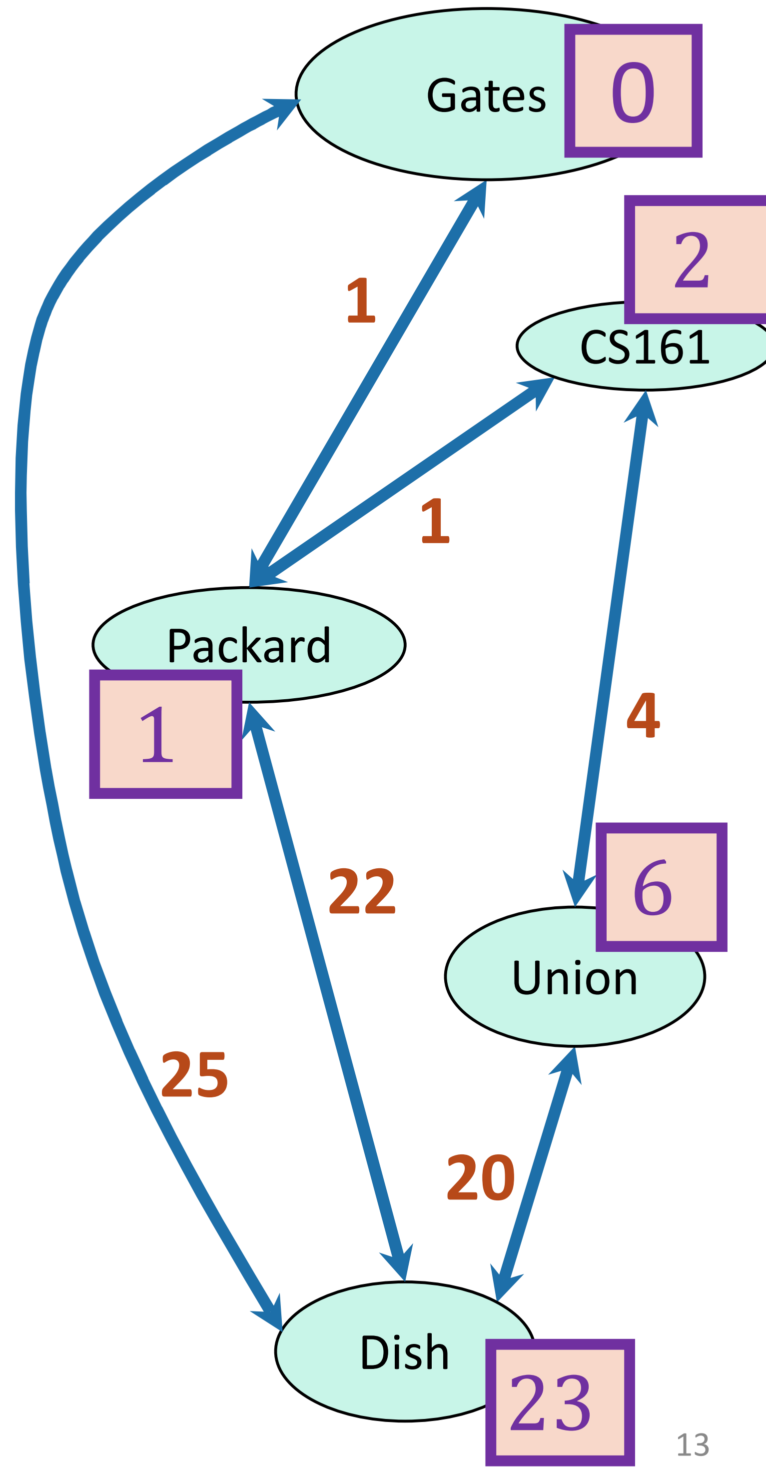


- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , d^{(i)}[u] + w(u,v))$
      where we are also taking the min over all u in v.inNeighbors

Interpretation of $d^{(i)}$

$d^{(i)}[v]$ is equal to the cost of the shortest path between s and v **with at most i edges**.

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$	0	1	2	**6**	23
$d^{(4)}$	0	1	2	6	23



Why does Bellman-Ford work?

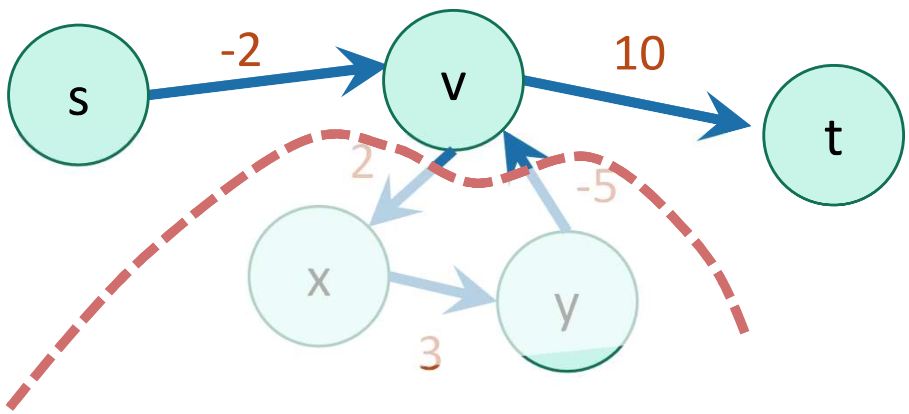
- Inductive hypothesis:
  - $d^{(i)}[v]$ is equal to the cost of the shortest path between s and v **with at most i edges**.
- Conclusion:
  - $d^{(n-1)}[v]$ is equal to the cost of the shortest path between s and v **with at most n-1 edges**.

Do the base case and inductive step!

Aside: simple paths

Assume there is no negative cycle.

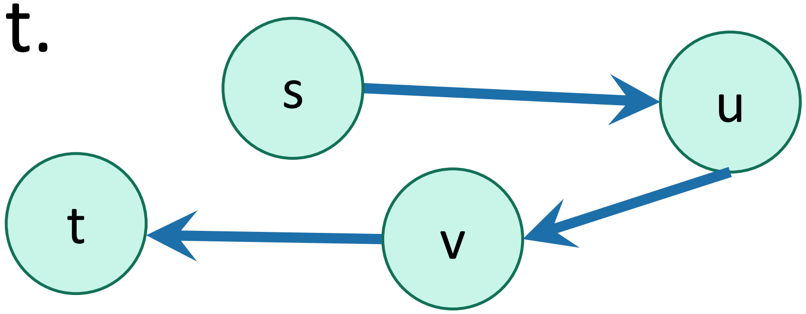
- Then there is a shortest path from s to t, and moreover there is a simple shortest path.



This cycle isn't helping.
Just get rid of it.

- A simple path in a graph with n vertices has at most n-1 edges in it.

Can't add another edge without making a cycle!



"Simple" means that the path has no cycles in it.

- So there is a shortest path with at most n-1 edges

Why does it work?

- Inductive hypothesis:
  - $d^{(i)}[v]$ is equal to the cost of the shortest path between s and v **with at most i edges**.
- Conclusion:
  - $d^{(n-1)}[v]$ is equal to the cost of the shortest path between s and v **with at most n-1 edges**.
  - **If there are no negative cycles**, $d^{(n-1)}[v]$ is equal to the cost of the shortest path.

Notice that negative edge weights are fine.
Just not negative cycles.

$G = (V,E)$ is a graph with n vertices and m edges.

Bellman-Ford algorithm

Bellman-Ford (G,s):

- Initialize arrays $d^{(0)},\ldots,d^{(n-1)}$ of length n
- $d^{(0)}[v] = \infty$ for all v in V
- $d^{(0)}[s] = 0$
- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(\, d^{(i)}[v]\, ,\, \min_{u \text{ in } v.\text{inNbrs}}\{d^{(i)}[u] + w(u,v)\}\,)$
- Now, dist(s,v) = $d^{(n-1)}[v]$ for all v in V.
  - (Assuming no negative cycles)

Here, Dijkstra picked a special vertex u and updated u's neighbors – Bellman-Ford will update all the vertices.

Note on implementation

- Don’t actually keep all n arrays around.
- Just keep two at a time: “last round” and “this round”

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$	0	1	2	**6**	23
$d^{(4)}$	0	1	2	6	23

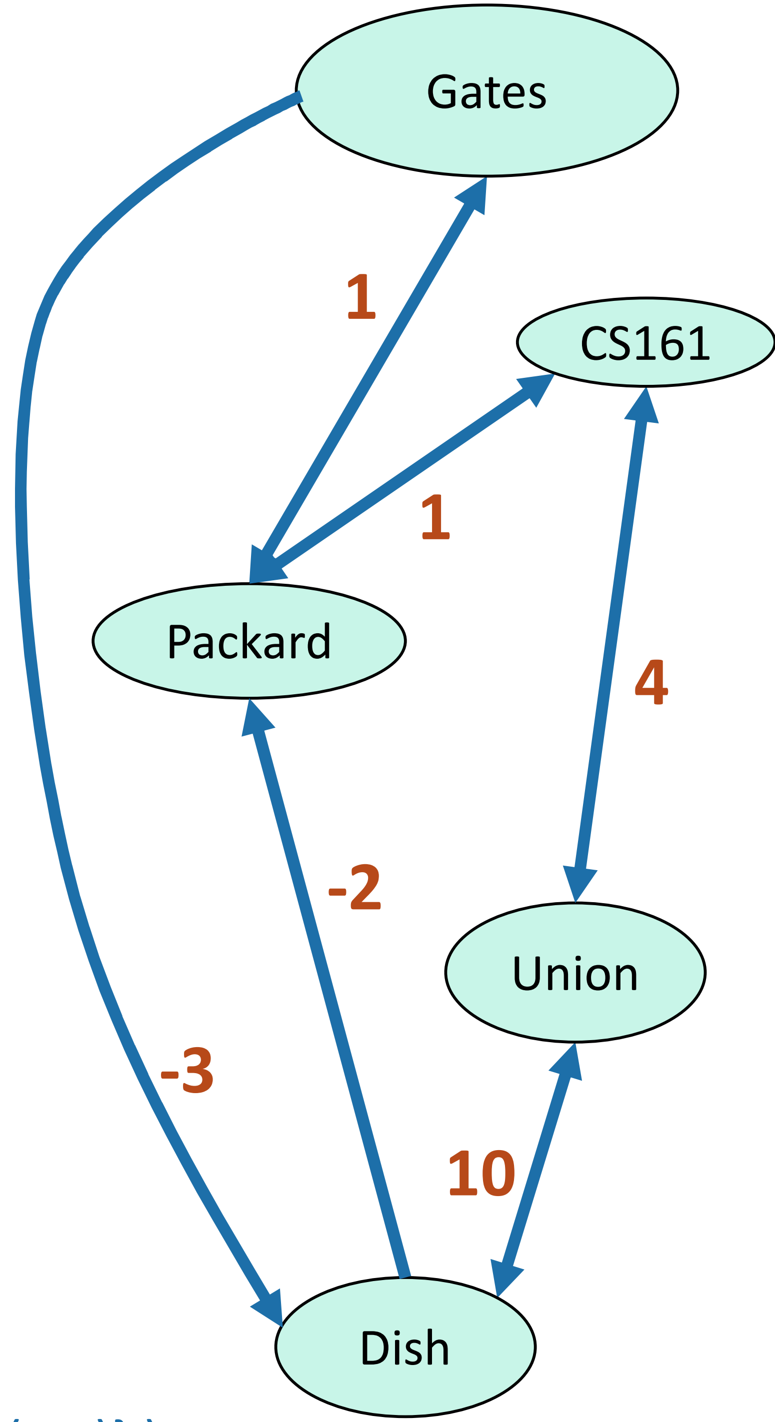
Only need these two in order to compute $d^{(4)}$

Một số đặc trưng của Bellman-Ford

- Running time is O(mn)
  - For each of n rounds, update m edges.
- Works fine with negative edges.
- Does not work with negative cycles.
  - No algorithm can – shortest paths aren't defined if there are negative cycles.
- B-F can detect negative cycles!

BF with negative cycles

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**-3**
$d^{(2)}$	0	**-5**	**2**	**7**	**-3**
$d^{(3)}$	**-4**	-5	**-4**	**6**	**-3**
$d^{(4)}$	-4	-5	-4	6	**-7**



- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , \min_{u \text{ in } v.nbrs}\{d^{(i)}[u] + w(u,v)\})$

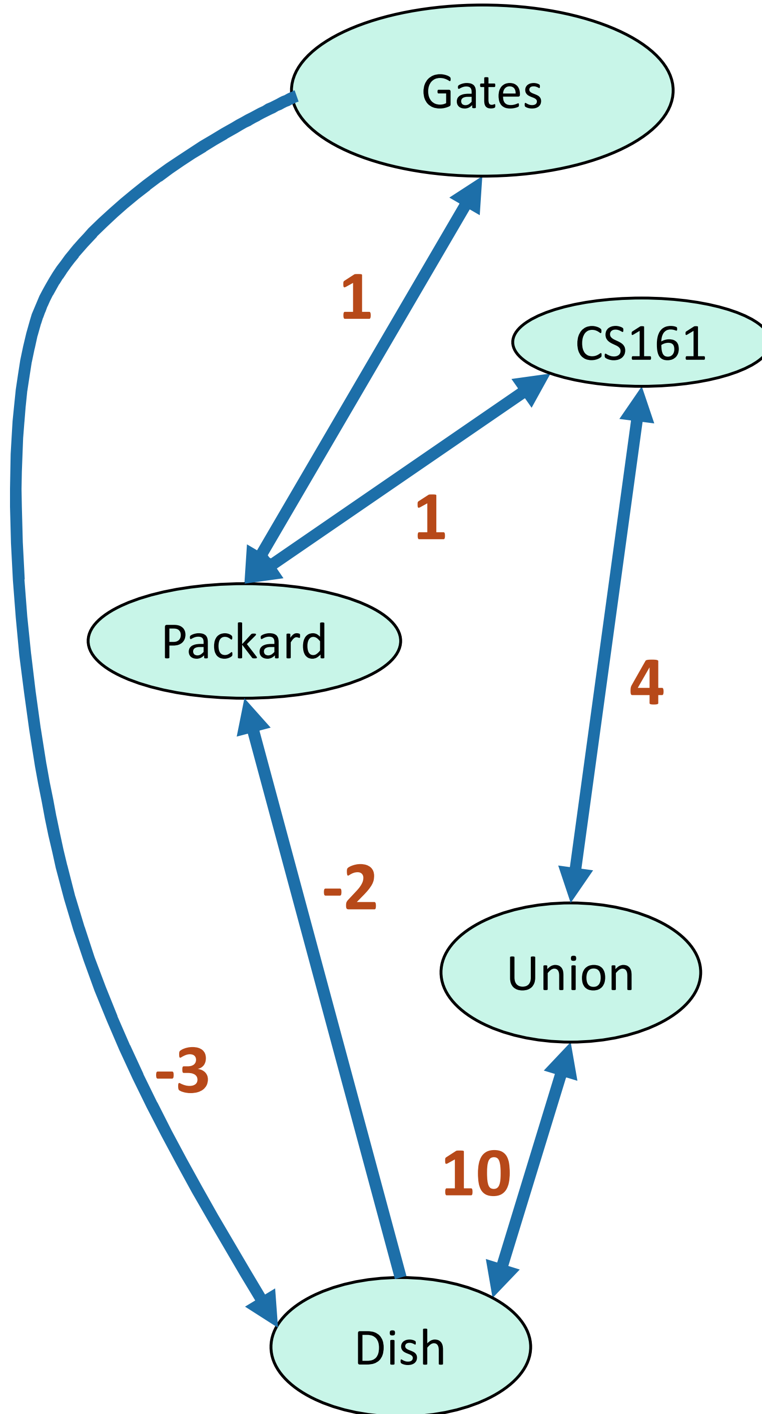
BF with negative cycles

	Gates	Packard	CS161	Union	Dish
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**-3**
$d^{(2)}$	0	**-5**	**2**	**7**	**-3**
$d^{(3)}$	**-4**	-5	**-4**	**6**	**-3**
$d^{(4)}$	-4	-5	-4	6	**-7**

But **we can tell** that it's not looking good:

	Gates	Packard	CS161	Union	Dish
$d^{(5)}$	-4	**-9**	-4	**3**	-7

Some stuff changed!



- **For** i=0,…,n-2:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , \min_{u \text{ in } v.nbrs}\{d^{(i)}[u] + w(u,v)\})$

Negative cycles in Bellman-Ford

- If there are no negative cycles:
  - Everything works as it should, and stabilizes in n-1 rounds.
- If there are negative cycles:
  - Not everything works as it should…
  - The d[v] values will keep changing.
- Solution:
  - Go one round more and see if things change.

Bellman-Ford algorithm

Bellman-Ford*(G,s):

- $d^{(0)}[v] = \infty$ for all v in V
- $d^{(0)}[s] = 0$
- **For** i=0,…,n-1:
  - **For** v in V:
    - $d^{(i+1)}[v] \leftarrow \min(d^{(i)}[v] , \min_{u \text{ in v.inNeighbors}} \{d^{(i)}[u] + w(u,v)\})$
- If $d^{(n-1)} != d^{(n)}$:
  - **Return** NEGATIVE CYCLE ☹
- Otherwise, $dist(s,v) = d^{(n-1)}[v]$

Running time: O(mn)

Bellman-Ford algorithm

- (-) Slower than Dijkstra's algorithm
- (+) Can handle negative edge weights.
  - Can **detect** negative cycles!
- (+) Allows for some flexibility if the weights change!?

Important thing about B-F
for the rest of this lecture

$d^{(i)}[v]$ is equal to the cost of the shortest path between s and v **with at most i edges**

B-F is an example of…
Dynamic Programming!

$d^{(i+1)}[v] \leftarrow \min(\, d^{(i)}[v] \,,\, \min_{u \text{ in } v.\text{inNbrs}}\{d^{(i)}[u] + w(u,v)\}\,)$

	Gates	Packard	C		
$d^{(0)}$	0	∞	∞	∞	∞
$d^{(1)}$	0	**1**	∞	∞	**25**
$d^{(2)}$	0	1	**2**	**45**	**23**
$d^{(3)}$	0	1	2	**6**	23
$d^{(4)}$	0	1	2	6	23