cs473: Algorithms

Lecture 4: Dynamic Programming

Michael A. Forbes

University of Illinois at Urbana-Champaign

September 9, 2019

Overview

Overview

logistics:

Overview

logistics:

- pset2 due R5

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*
- maximum independent set

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*
- maximum independent set
- dominating set

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*
- maximum independent set
- dominating set

today:

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*
- maximum independent set
- dominating set

today:

- shortest paths

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

- dynamic programming *on trees*
- maximum independent set
- dominating set

today:

- shortest paths
  - with negative lengths

Overview

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

last lecture:

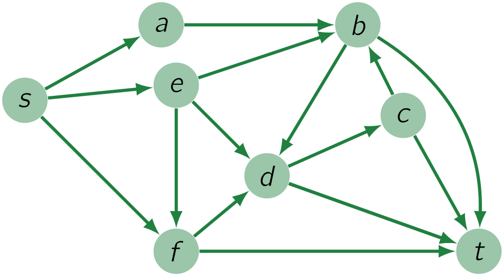
- dynamic programming *on trees*
- maximum independent set
- dominating set

today:

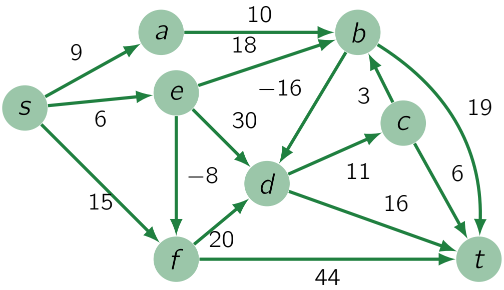
- shortest paths
  - with negative lengths
  - all-pairs

Shortest Paths, with Negative Lengths

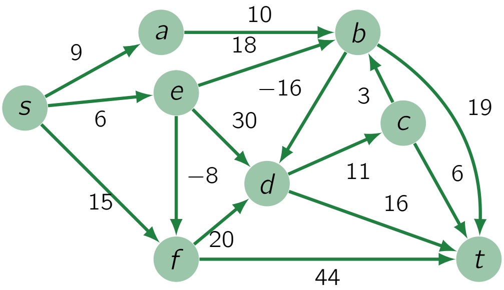
Shortest Paths, with Negative Lengths



Shortest Paths, with Negative Lengths

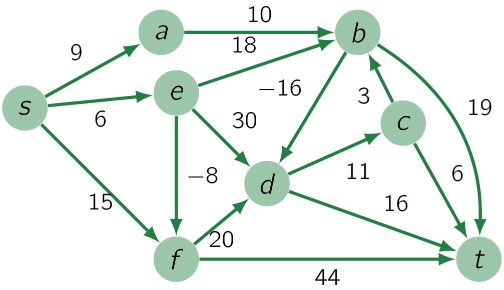


Shortest Paths, with Negative Lengths



questions:

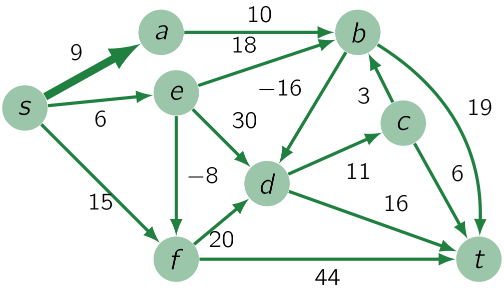
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?

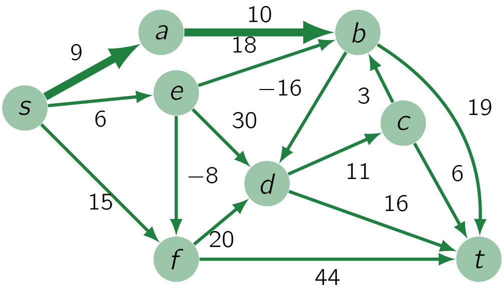
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?

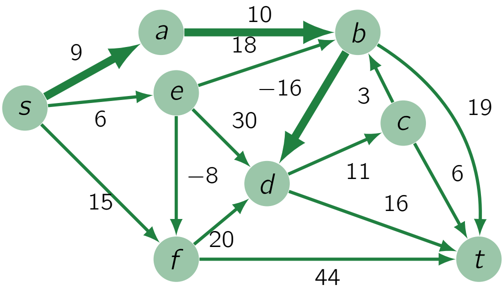
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?

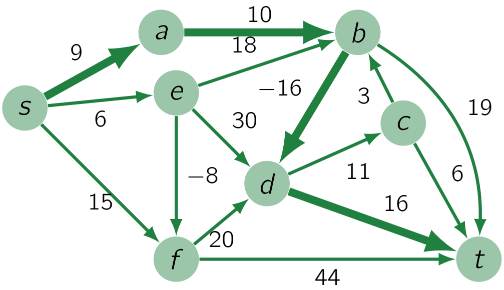
Shortest Paths, with Negative Lengths



questions:

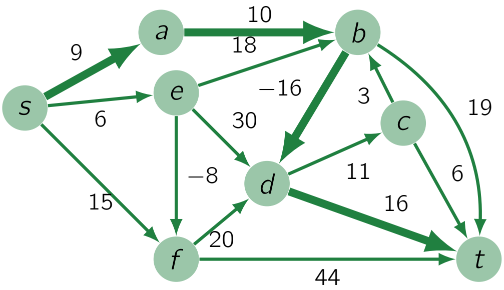
- what is the length of the shortest path between s and t?

Shortest Paths, with Negative Lengths



questions:

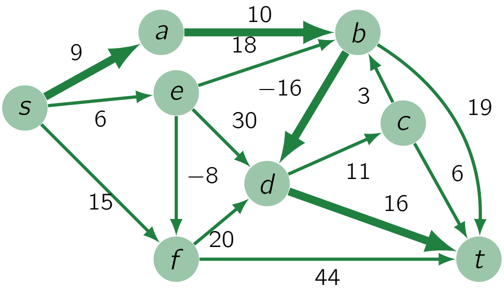
- what is the length of the shortest path between s and t?



questions:

- what is the length of the shortest path between s and t?

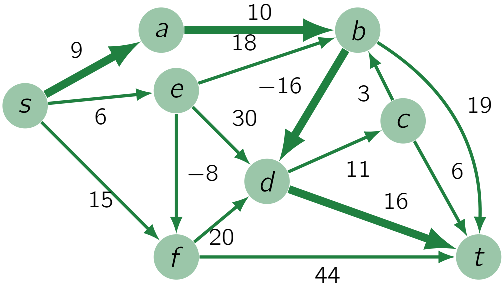
total cost:



questions:

- what is the length of the shortest path between s and t?

total cost: $9 + 10 + (-16) + 16 =$

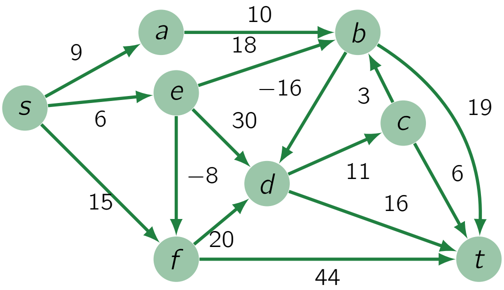


total cost: $9 + 10 + (-16) + 16 = 19$

questions:

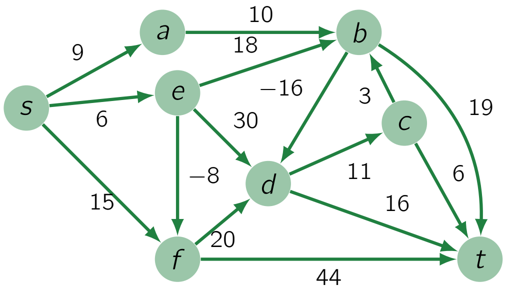
- what is the length of the shortest path between s and t?

Shortest Paths, with Negative Lengths



questions:

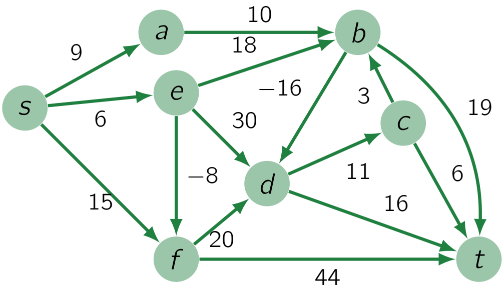
- what is the length of the shortest path between s and t?



questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?

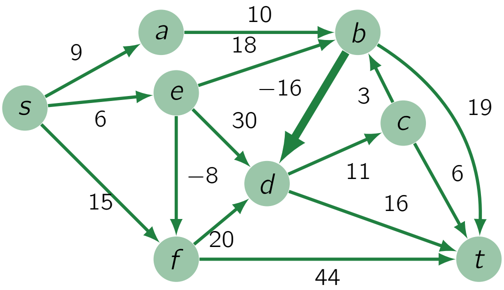
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

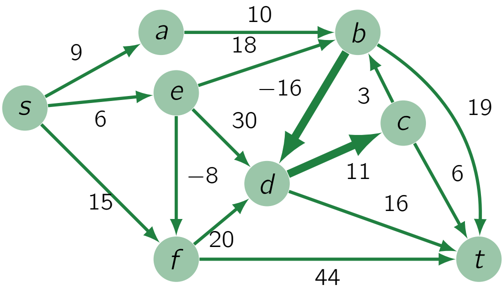
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

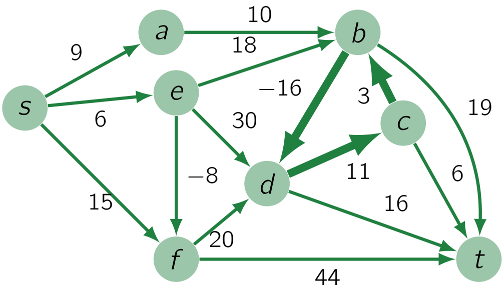
Shortest Paths, with Negative Lengths



questions:

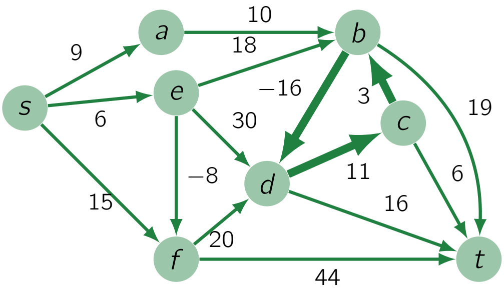
- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

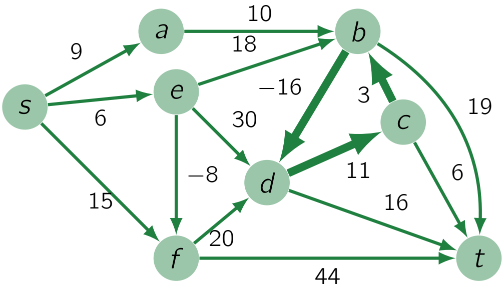


questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

total cost:

Shortest Paths, with Negative Lengths

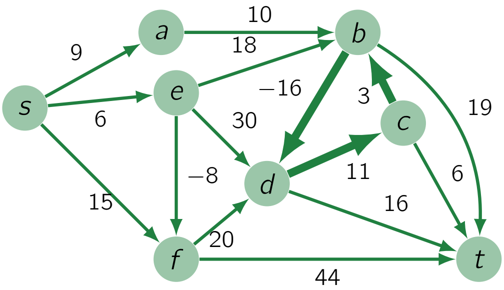


total cost: $-16 + 11 + 3 =$

questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

Shortest Paths, with Negative Lengths

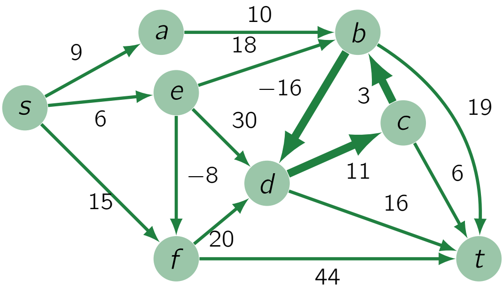


total cost: $-16 + 11 + 3 = -3$

questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?

Shortest Paths, with Negative Lengths

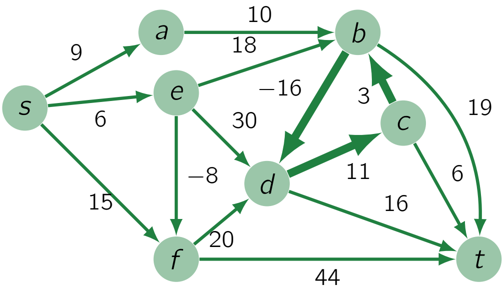


total cost: $-16 + 11 + 3 = -3$

questions:

- what is the length of the shortest path between *s* and *t*?
- what is the length of the shortest path from *s* to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

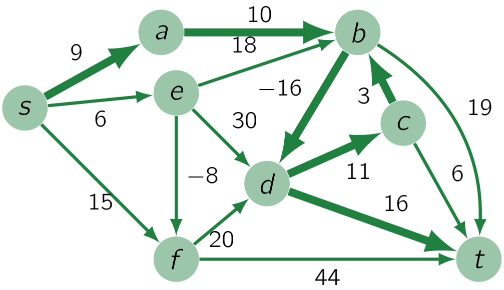
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

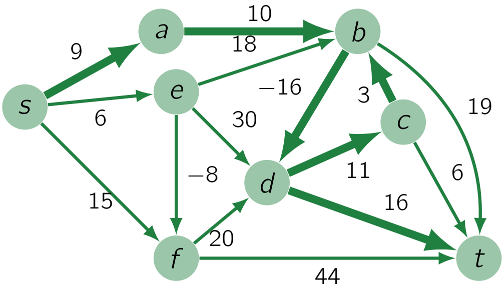
Shortest Paths, with Negative Lengths



questions:

- what is the length of the shortest path between *s* and *t*?
- what is the length of the shortest path from *s* to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

Shortest Paths, with Negative Lengths

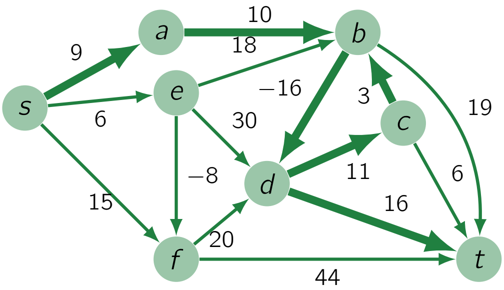


total cost:

questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

Shortest Paths, with Negative Lengths

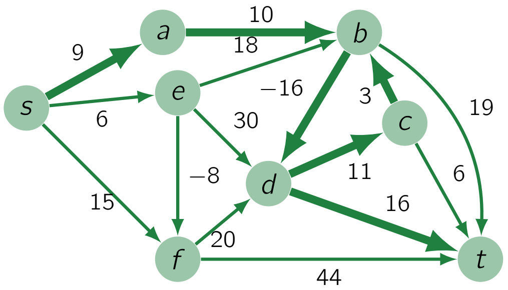


questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$9 + 10+$



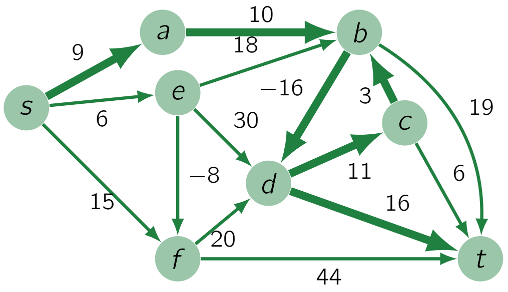
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$9 + 10 + (-16 + 11 + 3)$

Shortest Paths, with Negative Lengths



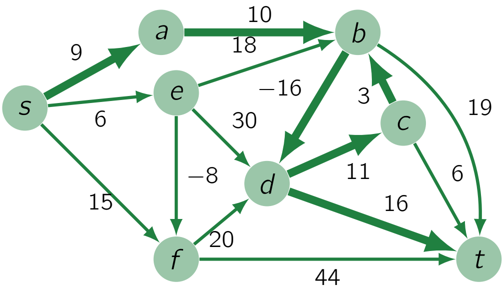
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k$$

Shortest Paths, with Negative Lengths

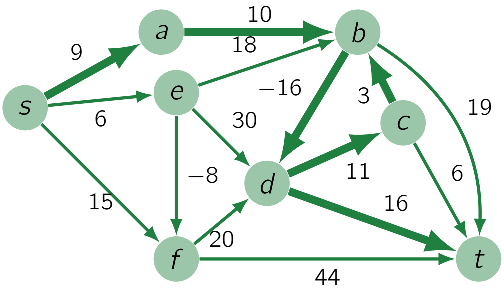


questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$



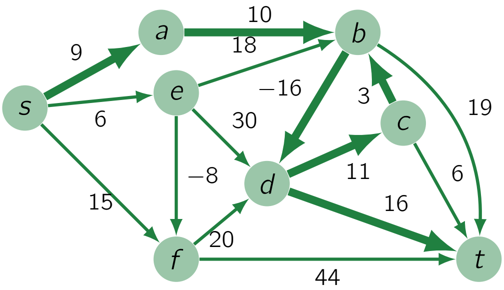
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k$$

Shortest Paths, with Negative Lengths



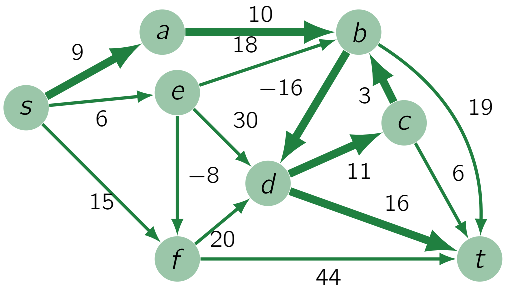
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

Shortest Paths, with Negative Lengths



total cost:

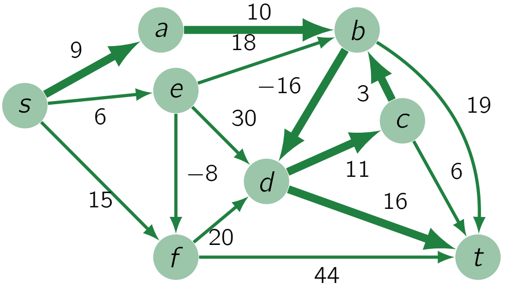
$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

remarks:

Shortest Paths, with Negative Lengths



total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

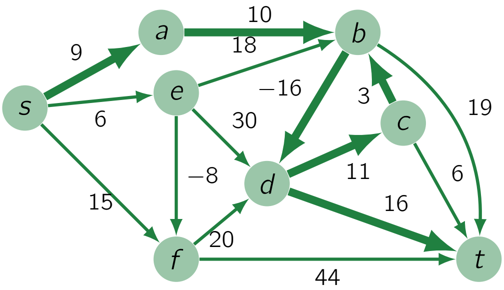
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

remarks:

- computing the length of the shortest *simple* $s \rightsquigarrow t$ path

Shortest Paths, with Negative Lengths



total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

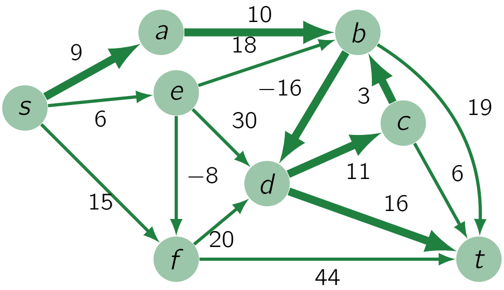
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

remarks:

- computing the length of the shortest *simple* $s \rightsquigarrow t$ path (with possibly negative lengths)

Shortest Paths, with Negative Lengths



total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

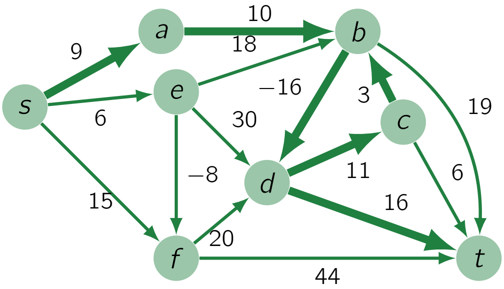
questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

remarks:

- computing the length of the shortest *simple* $s \rightsquigarrow t$ path (with possibly negative lengths) is NP-hard

Shortest Paths, with Negative Lengths



total cost:

$$9 + 10 + (-16 + 11 + 3) \cdot k + (-16) + 16$$
$$= 19 - 3k \to -\infty$$

questions:

- what is the length of the shortest path between s and t?
- what is the length of the shortest path from s to *every* other node?
- what happens if we get lost?
- how to deal with *negative cycles*?

remarks:

- computing the length of the shortest *simple* $s \rightsquigarrow t$ path (with possibly negative lengths) is NP-hard — contains the *Hamiltonian path* problem

Shortest Paths, with Negative Lengths (II)

Shortest Paths, with Negative Lengths (II)

Definition

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph,

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G,

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\mathrm{dist}(s,t)$,

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\mathrm{dist}(s, t)$, is the length of the shortest (s, t)-walk,

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s, t)$, is the length of the shortest (s, t)-walk, $\text{dist}(s, t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s,t)$, is the length of the shortest (s,t)-walk, $\text{dist}(s,t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s, t)$, is the length of the shortest (s, t)-walk, $\text{dist}(s, t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s, t)-walk containing a negative length cycle

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\mathrm{dist}(s,t)$, is the length of the shortest (s,t)-walk, $\mathrm{dist}(s,t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s,t)-walk containing a negative length cycle $\implies \mathrm{dist}(s,t) = -\infty$

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s,t)$, is the length of the shortest (s,t)-walk, $\text{dist}(s,t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s,t)-walk containing a negative length cycle $\implies$ $\text{dist}(s,t) = -\infty$
- *no* (s,t)-walk containing a negative length cycle

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s, t)$, is the length of the shortest (s, t)-walk, $\text{dist}(s, t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s, t)-walk containing a negative length cycle $\implies$ $\text{dist}(s, t) = -\infty$
- *no* (s, t)-walk containing a negative length cycle $\implies$ shortest walk is a *path*

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \ldots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s, t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s, t)$, is the length of the shortest (s, t)-walk, $\text{dist}(s, t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s, t)-walk containing a negative length cycle $\implies$ $\text{dist}(s, t) = -\infty$
- *no* (s, t)-walk containing a negative length cycle $\implies$ shortest walk is a *path* $\implies$ shortest walk $\leq n - 1$ edges

Shortest Paths, with Negative Lengths (II)

Definition

$G = (V, E)$ *directed* (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

- A **path in** G is a sequence of *distinct* vertices $v_0, v_1, \dots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-path is a path where $v_0 = s$ and $v_k = t$.
- A **walk in** G is a sequence of vertices $v_0, v_1, \dots, v_k \in V$ such that $(v_i, v_{i+1}) \in E$ for all i. An (s,t)-walk is a walk where $v_0 = s$ and $v_k = t$.
- The **length of a walk** is the sum of the edge lengths $\sum_i \ell(v_i, v_{i+1})$.
- The **distance from** s **to** t **in** G, denoted $\text{dist}(s,t)$, is the length of the shortest (s,t)-walk, $\text{dist}(s,t) := \min_{(s,t)\text{-walk } w} \ell(w)$.

remarks:

- (s,t)-walk containing a negative length cycle $\implies$ $\text{dist}(s,t) = -\infty$
- *no* (s,t)-walk containing a negative length cycle $\implies$ shortest walk is a *path* $\implies$ shortest walk $\leq n-1$ edges and is of finite length

Shortest Paths, with Negative Lengths (III)

Shortest Paths, with Negative Lengths (III)

Definition

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$,

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \rightarrow \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \text{dist}(s, t) = -\infty$)

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \text{dist}(s, t) = -\infty$)
- given $s \in V$,

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies$ $\text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies$ $\text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \mathrm{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\mathrm{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

- negative lengths can be natural in modelling real life

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \mathrm{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\mathrm{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

- negative lengths can be natural in modelling real life
  - e.g., demand/supply on an electrical grid,

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

- negative lengths can be natural in modelling real life
  - e.g., demand/supply on an electrical grid, negative cycles manifest as *arbitrage*

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies \text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

- negative lengths can be natural in modelling real life
    - e.g., demand/supply on an electrical grid, negative cycles manifest as *arbitrage*
- negative lengths can arise as by-products of other algorithms,

Shortest Paths, with Negative Lengths (III)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. The **(single-source) shortest path problem (with negative weights)** is to:

- given $s, t \in V$, find a minimum length (s, t)-path or find an (s, t)-walk with a negative cycle ($\implies$ $\text{dist}(s, t) = -\infty$)
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$
- determine if G has *any* negative cycle

remarks:

- negative lengths can be natural in modelling real life
  - e.g., demand/supply on an electrical grid, negative cycles manifest as *arbitrage*
- negative lengths can arise as by-products of other algorithms, e.g., flows in graphs

Dijkstra's Algorithm

Dijkstra's Algorithm

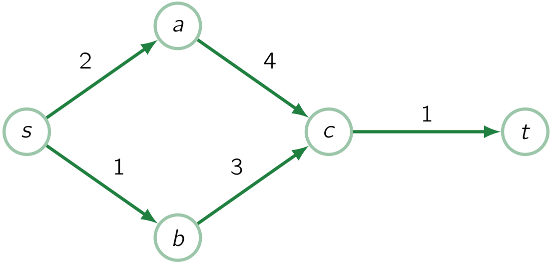
Dijkstra's algorithm:

Dijkstra's Algorithm

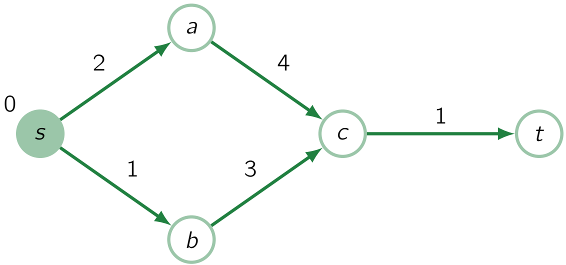
Dijkstra's algorithm: greedily grow shortest paths from source s

Dijkstra's Algorithm

Dijkstra's algorithm: greedily grow shortest paths from source s

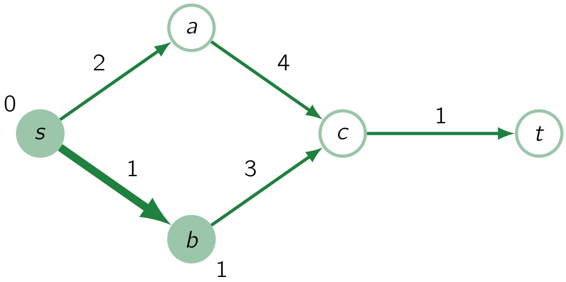


Dijkstra's algorithm: greedily grow shortest paths from source s



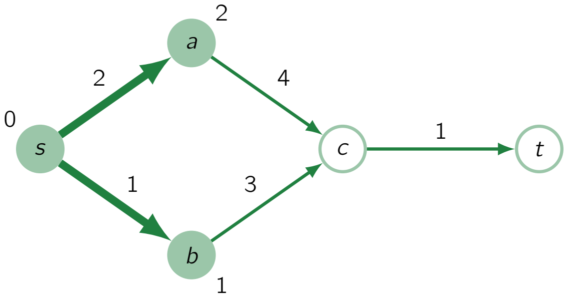
Dijkstra's Algorithm

Dijkstra's algorithm: greedily grow shortest paths from source s

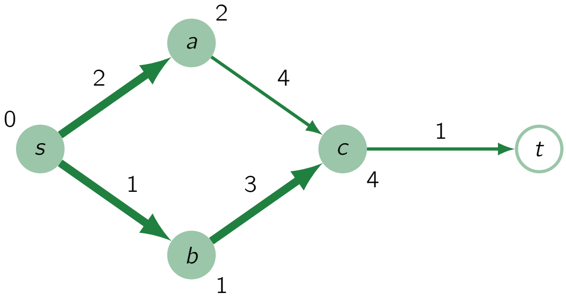


Dijkstra's Algorithm

Dijkstra's algorithm: greedily grow shortest paths from source s

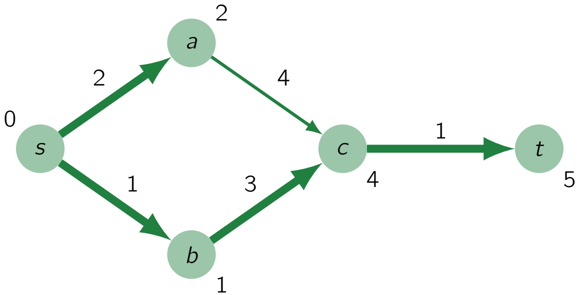


Dijkstra's algorithm: greedily grow shortest paths from source *s*



Dijkstra's Algorithm

Dijkstra's algorithm: greedily grow shortest paths from source *s*



Dijkstra's Algorithm,

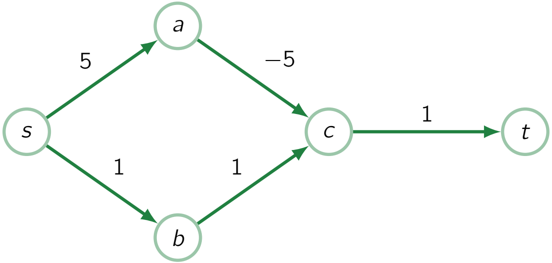
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s

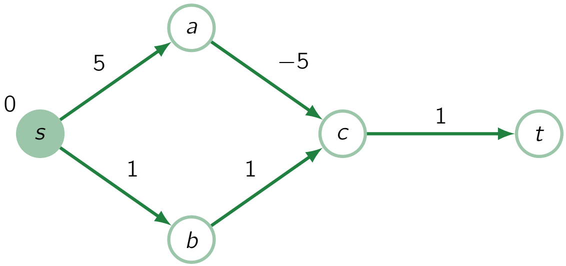
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



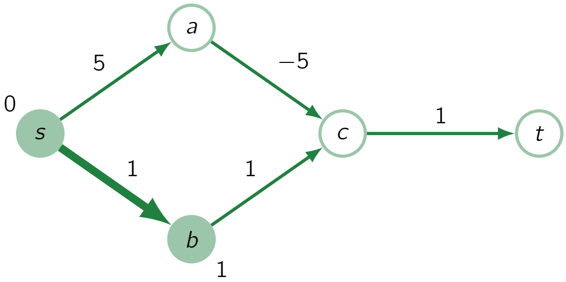
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



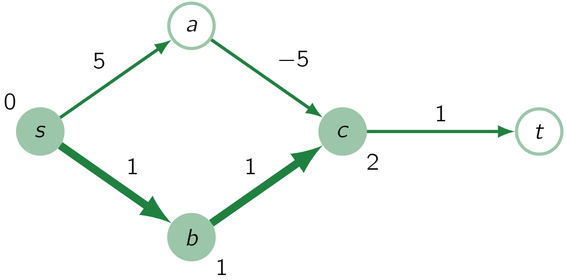
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



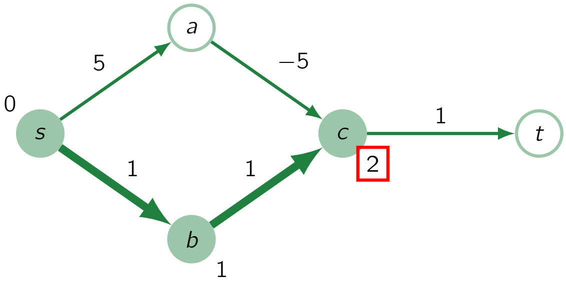
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



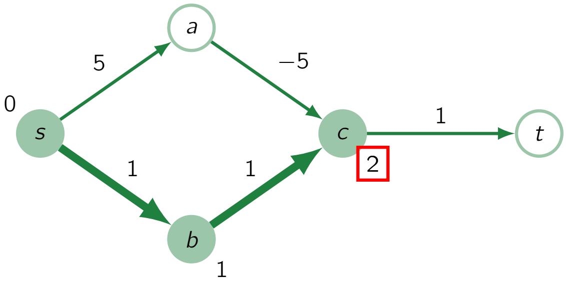
Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



Dijkstra's Algorithm, with Negative Lengths?

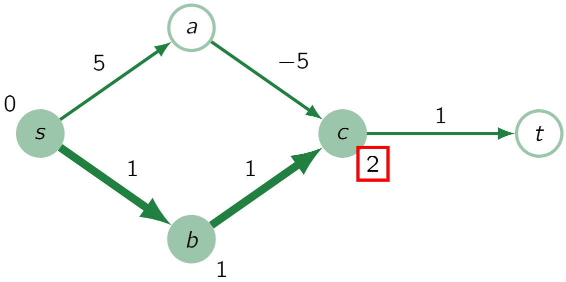
Dijkstra's algorithm: greedily grow shortest paths from source s



remarks:

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s

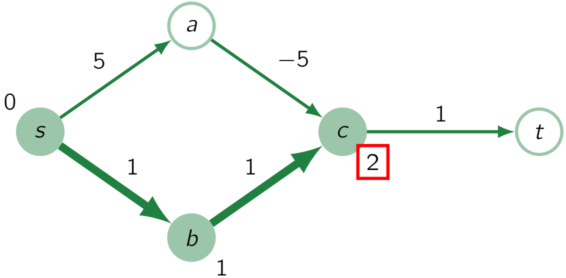


remarks:

- greedy exploration,

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s

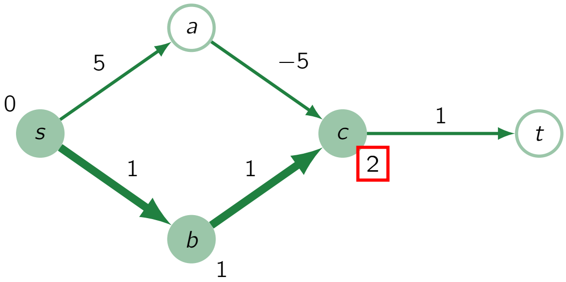


remarks:

- greedy exploration, ordering vertices $v \in V$ by $\text{dist}(s, v)$

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s

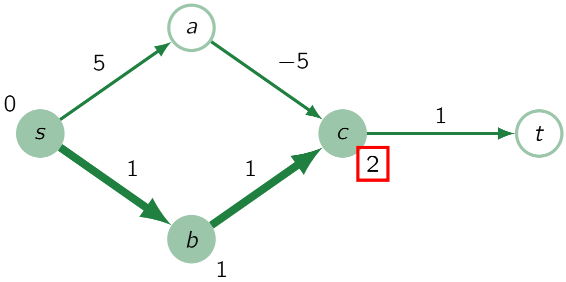


remarks:

- greedy exploration, ordering vertices $v \in V$ by $\mathrm{dist}(s, v)$ — without updates!

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



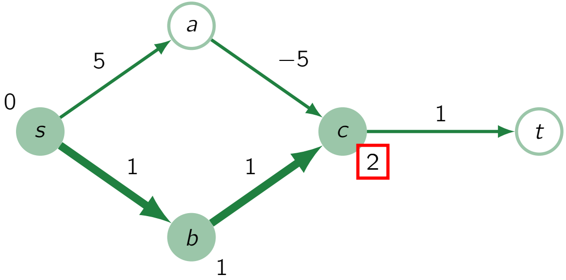
remarks:

- greedy exploration, ordering vertices $v \in V$ by $\mathrm{dist}(s, v)$ — without updates!

$\Longrightarrow$ algorithm assumes the distance only grows as the graph is explored

Dijkstra's Algorithm, with Negative Lengths?

Dijkstra's algorithm: greedily grow shortest paths from source s



remarks:

- greedy exploration, ordering vertices $v \in V$ by $\mathrm{dist}(s, v)$ — without updates!

$\Longrightarrow$ algorithm assumes the distance only grows as the graph is explored

$\equiv$ assumes all edge lengths are non-negative

Shortest Paths, with Negative Lengths (IV)

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If
$s = v_0$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If
$s = v_0 \to v_1$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If
$s = v_0 \to v_1 \to v_2$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If
$s = v_0 \to v_1 \to v_2 \to \cdots$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If
$s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk,

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s,t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk,*

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,*

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \le k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \le \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste.

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\mathrm{dist}(s, v_i) \leq \mathrm{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\mathrm{dist}(s, v_i) \leq \mathrm{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths,

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle
- Dijkstra's algorithm defines subproblems

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle
- Dijkstra's algorithm defines subproblems by restricting the graph by $\text{dist}(s, \cdot)$

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative, $\mathrm{dist}(s, v_i) \leq \mathrm{dist}(s, v_{i+1})$ for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle
- Dijkstra's algorithm defines subproblems by restricting the graph by $\mathrm{dist}(s, \cdot)$
- *idea:*

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle
- Dijkstra's algorithm defines subproblems by restricting the graph by $\text{dist}(s, \cdot)$
- *idea:* parameterize subproblems by *number* of edges in a walk,

Shortest Paths, with Negative Lengths (IV)

Lemma

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. If $s = v_0 \to v_1 \to v_2 \to \cdots \to v_k = t$ is a shortest (s, t)-walk, then

1. *$s \to v_1 \to \cdots \to v_i$ is a shortest (s, v_i)-walk, for $i \leq k$*
2. **if** *ℓ is non-negative,* $\text{dist}(s, v_i) \leq \text{dist}(s, v_{i+1})$ *for all i*

Proof.

(1) Cut and paste. (2) Clear. □

remarks:

- shortest walks *are* shortest paths, if no negative cycle
- Dijkstra's algorithm defines subproblems by restricting the graph by $\text{dist}(s, \cdot)$
- *idea:* parameterize subproblems by *number* of edges in a walk, *and* allow updates to $\text{dist}(s, \cdot)$

Shortest Paths, with Negative Lengths (V)

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\text{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\leq k$ *edges*.

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\text{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\leq k$ *edges*.

$$\text{dist}_k(s,t) := \min_{\substack{(s,t)\text{-walk } w \\ |w| \leq k}} \ell(w) \; .$$

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\text{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\leq k$ *edges*.

$$\text{dist}_k(s, t) := \min_{\substack{(s,t)\text{-walk } w \\ |w| \leq k}} \ell(w) \ .$$

remarks:

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\mathrm{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\leq k$ *edges*.

$$\mathrm{dist}_k(s,t) := \min_{\substack{(s,t)\text{-walk } w \\ |w| \leq k}} \ell(w) \ .$$

remarks:

- $\mathrm{dist}_k(s, t) = \infty$ if no $(\leq k)$-edge (s, t)-walk

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\mathrm{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\le k$ *edges*.

$$\mathrm{dist}_k(s,t) := \min_{\substack{(s,t)\text{-walk } w \\ |w| \le k}} \ell(w) \ .$$

remarks:

- $\mathrm{dist}_k(s, t) = \infty$ if no $(\le k)$-edge (s, t)-walk
- $\mathrm{dist}_0(s, s) = 0$,

Shortest Paths, with Negative Lengths (V)

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. For $s, t \in V$, define $\text{dist}_k(s, t)$ to be the length of the shortest (s, t)-walk *using* $\leq k$ *edges*.

$$\text{dist}_k(s, t) := \min_{\substack{(s,t)\text{-walk } w \\ |w| \leq k}} \ell(w) \, .$$

remarks:

- $\text{dist}_k(s, t) = \infty$ if no $(\leq k)$-edge (s, t)-walk
- $\text{dist}_0(s, s) = 0$, $\text{dist}_0(s, v) = \infty$ for $v \neq s$

Shortest Paths, with Negative Lengths (VI)

Lemma

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$,

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$.

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\text{dist}_k(s, t) =$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s, t) = \min \left\{ \right.$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \end{cases}$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \end{cases}$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \end{cases}$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \leq k$ (s,t)-walk.

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \leq k$ (s,t)-walk. Then,

- $j < k$:

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \le k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\le k-1)$-edge (s,t)-walk of length $\mathrm{dist}_{k-1}(s,t)$

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \leq k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\leq k-1)$-edge (s,t)-walk of length $\text{dist}_{k-1}(s,t)$
- $j = k$:

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \leq k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\leq k-1)$-edge (s,t)-walk of length $\mathrm{dist}_{k-1}(s,t)$
- $j = k$: hence $s = v_0 \to v_1 \to v_2 \to \cdots \to v_{k-1}$ is a shortest length $(\leq k-1)$-edge (s, v_{k-1}) walk

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \leq k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\leq k-1)$-edge (s,t)-walk of length $\text{dist}_{k-1}(s,t)$
- $j = k$: hence $s = v_0 \to v_1 \to v_2 \to \cdots \to v_{k-1}$ is a shortest length $(\leq k-1)$-edge (s, v_{k-1}) walk $\implies$ can add $\ell(v_{k-1}, t)$ to reach t □

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \le k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\le k-1)$-edge (s,t)-walk of length $\mathrm{dist}_{k-1}(s,t)$
- $j = k$: hence $s = v_0 \to v_1 \to v_2 \to \cdots \to v_{k-1}$ is a shortest length $(\le k-1)$-edge (s, v_{k-1}) walk $\implies$ can add $\ell(v_{k-1}, t)$ to reach t □

remark:

Shortest Paths, with Negative Lengths (VI)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. *Then for all* $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V} \{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Proof.

Let $s = v_0 \to v_1 \to v_2 \to \cdots \to v_j = t$ be a shortest length $j \le k$ (s,t)-walk. Then,

- $j < k$: hence this is a $(\le k-1)$-edge (s,t)-walk of length $\text{dist}_{k-1}(s,t)$
- $j = k$: hence $s = v_0 \to v_1 \to v_2 \to \cdots \to v_{k-1}$ is a shortest length $(\le k-1)$-edge (s, v_{k-1}) walk $\implies$ can add $\ell(v_{k-1}, t)$ to reach t □

remark: $\ell(v,t) = \infty$ if there is no edge

Shortest Paths, with Negative Lengths (VII)

Shortest Paths, with Negative Lengths (VII)

Theorem

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$,

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$,

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s.

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles,*

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles, then for all $v \in V$,*

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles, then for all $v \in V$,* $\text{dist}_{n-1}(s, v)$

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles, then for all $v \in V$,* $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$,

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*.*

1. *If there are no negative length cycles, then for all* $v \in V$*,* $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$*, and even* $\mathrm{dist}_{n-1}(s, v) = \mathrm{dist}(s, v)$*.*

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*.*

1. *If there are no negative length cycles, then for all* $v \in V$*,* $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$*, and even* $\text{dist}_{n-1}(s, v) = \text{dist}(s, v)$*.*
2. *If for all* $v \in V$*,* $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$*,*

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles, then for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, and even $\text{dist}_{n-1}(s, v) = \text{dist}(s, v)$.*
2. *If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, then there are no negative length cycles.*

Shortest Paths, with Negative Lengths (VIII)

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\text{dist}_k(s,v) \leq \text{dist}_{k-1}(s,v)$*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\text{dist}_k(s,v) \leq \text{dist}_{k-1}(s,v)$*
- *If all $v \in V$,*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\mathrm{dist}_k(s,v) \leq \mathrm{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\mathrm{dist}_k(s,v) = \mathrm{dist}_{k-1}(s,v)$*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\text{dist}_k(s,v) \leq \text{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\text{dist}_k(s,v) = \text{dist}_{k-1}(s,v)$*

$\Longrightarrow$ *all $v \in V$,*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\text{dist}_k(s,v) \leq \text{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\text{dist}_k(s,v) = \text{dist}_{k-1}(s,v)$*

$\Longrightarrow$ *all $v \in V$, $\text{dist}_{k+1}(s,v)$*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\mathrm{dist}_k(s,v) \leq \mathrm{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\mathrm{dist}_k(s,v) = \mathrm{dist}_{k-1}(s,v)$*

$\implies$ *all $v \in V$, $\mathrm{dist}_{k+1}(s,v) = \mathrm{dist}_k(s,v)$*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\mathrm{dist}_k(s,t) = \min \begin{cases} \mathrm{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\mathrm{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\mathrm{dist}_k(s,v) \leq \mathrm{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\mathrm{dist}_k(s,v) = \mathrm{dist}_{k-1}(s,v)$*

$\Longrightarrow$ *all $v \in V$, $\mathrm{dist}_{k+1}(s,v) = \mathrm{dist}_k(s,v)$*

$\Longrightarrow$ *all $v \in V$, $\mathrm{dist}_{k+2}(s,v) = \mathrm{dist}_{k+1}(s,v)$*

Shortest Paths, with Negative Lengths (VIII)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$. Then for all $s, t \in V$,

$$\text{dist}_k(s,t) = \min \begin{cases} \text{dist}_{k-1}(s,t) \\ \min_{v \in V}\{\text{dist}_{k-1}(s,v) + \ell(v,t)\} \end{cases} .$$

Corollary

For all $k \geq 0$,

- *all $v \in V$, $\text{dist}_k(s,v) \leq \text{dist}_{k-1}(s,v)$*
- *If all $v \in V$, $\text{dist}_k(s,v) = \text{dist}_{k-1}(s,v)$*

$\Longrightarrow$ *all $v \in V$, $\text{dist}_{k+1}(s,v) = \text{dist}_k(s,v)$*

$\Longrightarrow$ *all $v \in V$, $\text{dist}_{k+2}(s,v) = \text{dist}_{k+1}(s,v) \Longrightarrow \cdots$*

Shortest Paths, with Negative Lengths (IX)

Proposition

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\text{dist}_{n-1}(s, v)$

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \le \mathrm{dist}_n(s, v)$.

Proof.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\text{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \text{dist}_{n-1}(s, v)$.
- If $k = n$,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated $\equiv$ there is a cycle.

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated $\equiv$ there is a cycle. As the cycle is of non-negative length $C \geq 0$,

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\text{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \text{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated $\equiv$ there is a cycle. As the cycle is of non-negative length $C \geq 0$, we can remove it to obtain a $(< n)$-edge (s, v)-walk of value $d = \text{dist}_n(s, v) - C$

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\operatorname{dist}_{n-1}(s, v) \leq \operatorname{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\operatorname{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \operatorname{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated $\equiv$ there is a cycle. As the cycle is of non-negative length $C \geq 0$, we can remove it to obtain a $(< n)$-edge (s, v)-walk of value $d = \operatorname{dist}_n(s, v) - C$ with $\operatorname{dist}_n(s, v) \geq d$

Shortest Paths, with Negative Lengths (IX)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$.

Proof.

Let $s = v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k = v$ be a walk of $(\leq n)$-edges, with length $\mathrm{dist}_n(s, v)$.

- If $k < n$, then this is a $(< n)$-edge walk and hence of length $\geq \mathrm{dist}_{n-1}(s, v)$.
- If $k = n$, then the walk visits $n + 1$ vertices $\implies$ some vertex is repeated $\equiv$ there is a cycle. As the cycle is of non-negative length $C \geq 0$, we can remove it to obtain a $(< n)$-edge (s, v)-walk of value $d = \mathrm{dist}_n(s, v) - C$ with $\mathrm{dist}_n(s, v) \geq d \geq \mathrm{dist}_{n-1}(s, v)$. □

Shortest Paths, with Negative Lengths (X)

Shortest Paths, with Negative Lengths (X)

Proposition

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s.

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$,
$\mathrm{dist}_{n-1}(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$,
$\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$,

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\operatorname{dist}_{n-1}(s, v) \leq \operatorname{dist}_n(s, v)$, then $\lim_{k\to\infty} \operatorname{dist}_k(s, v)$ is finite for all $v \in V$.

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary,

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, then $\lim_{k\to\infty} \text{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\text{dist}_{n-1}(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, then $\lim_{k\to\infty} \text{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v) \implies$ for all $v \in V$, $\text{dist}_{n-1}(s, v) = \text{dist}_n(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, then $\lim_{k\to\infty} \text{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v) \implies$ for all $v \in V$, $\text{dist}_{n-1}(s, v) = \text{dist}_n(s, v) = \text{dist}_{n+1}(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, then $\lim_{k\to\infty} \text{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v) \implies$ for all $v \in V$, $\text{dist}_{n-1}(s, v) = \text{dist}_n(s, v) = \text{dist}_{n+1}(s, v) = \text{dist}_{n+2}(s, v)$

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v) \implies$ for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) = \mathrm{dist}_n(s, v) = \mathrm{dist}_{n+1}(s, v) = \mathrm{dist}_{n+2}(s, v) = \cdots$.

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v) \implies$ for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) = \mathrm{dist}_n(s, v) = \mathrm{dist}_{n+1}(s, v) = \mathrm{dist}_{n+2}(s, v) = \cdots$. As all v are reachable from s

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v) \implies$ for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) = \mathrm{dist}_n(s, v) = \mathrm{dist}_{n+1}(s, v) = \mathrm{dist}_{n+2}(s, v) = \cdots$. As all v are reachable from $s \implies -\infty < \mathrm{dist}_{n-1}(s, v) < \infty$ for all k and v.

Shortest Paths, with Negative Lengths (X)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ is finite for all $v \in V$.

Proof.

By previous corollary, for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v) \implies$ for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) = \mathrm{dist}_n(s, v) = \mathrm{dist}_{n+1}(s, v) = \mathrm{dist}_{n+2}(s, v) = \cdots$. As all v are reachable from $s \implies -\infty < \mathrm{dist}_{n-1}(s, v) < \infty$ for all k and v. Hence $\lim_{k\to\infty} \mathrm{dist}_k(s, v) = \mathrm{dist}_{n-1}(s, v)$ is finite for all v. □

Shortest Paths, with Negative Lengths (XI)

Shortest Paths, with Negative Lengths (XI)

Proposition

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle,

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v) = -\infty$.

Proof.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L,

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L, where $u \rightsquigarrow u$ is a negative length cycle of length $-C < 0$.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L, where $u \rightsquigarrow u$ is a negative length cycle of length $-C < 0$. Then consider the (s, v)-walk $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$,

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L, where $u \rightsquigarrow u$ is a negative length cycle of length $-C < 0$. Then consider the (s, v)-walk $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$, which is of value $L - C$.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \mathrm{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L, where $u \rightsquigarrow u$ is a negative length cycle of length $-C < 0$. Then consider the (s, v)-walk $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$, which is of value $L - C$. Hence, for any j there is (s, v)-walk of length $L - C \cdot j$.

Shortest Paths, with Negative Lengths (XI)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Proof.

Let $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$ be an (s, v)-walk with length L, where $u \rightsquigarrow u$ is a negative length cycle of length $-C < 0$. Then consider the (s, v)-walk $s \rightsquigarrow u \rightsquigarrow u \rightsquigarrow u \rightsquigarrow v$, which is of value $L - C$. Hence, for any j there is (s, v)-walk of length $L - C \cdot j$. Hence $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$. □

Shortest Paths, with Negative Lengths (XII)

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If for all* $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ *is finite for all* $v \in V$.

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If for all* $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, $\lim_{k\to\infty} \mathrm{dist}_k(s, v)$ *is finite for all* $v \in V$*.*

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If there is a* (s, v)*-walk containing a negative length cycle, then* $\lim_{k\to\infty} \mathrm{dist}_k(s, v) = -\infty$.

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If for all $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, $\lim_{k\to\infty} \text{dist}_k(s, v)$ is finite for all $v \in V$.

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s. If there is a (s, v)-walk containing a negative length cycle, then $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Corollary

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s. *If for all* $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, $\lim_{k\to\infty} \text{dist}_k(s, v)$ *is finite for all* $v \in V$.

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s. *If there is a* (s, v)*-walk containing a negative length cycle, then* $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Corollary

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s.

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s. *If for all* $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, $\lim_{k\to\infty} \text{dist}_k(s, v)$ *is finite for all* $v \in V$.

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s. *If there is a* (s, v)*-walk containing a negative length cycle, then* $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$.

Corollary

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s. *If for all* $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$,

Shortest Paths, with Negative Lengths (XII)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If for all* $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, $\lim_{k\to\infty} \text{dist}_k(s, v)$ *is finite for all* $v \in V$*.*

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If there is a* (s, v)*-walk containing a negative length cycle, then* $\lim_{k\to\infty} \text{dist}_k(s, v) = -\infty$*.*

Corollary

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, *with every vertex reachable from* s*. If for all* $v \in V$, $\text{dist}_{n-1}(s, v) \leq \text{dist}_n(s, v)$, *then there are no negative length cycles.*

Shortest Paths, with Negative Lengths (VII)

Shortest Paths, with Negative Lengths (VII)

Theorem

Shortest Paths, with Negative Lengths (VII)

Theorem

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, $s \in V$, with every vertex reachable from s.

1. *If there are no negative length cycles, then for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, and $\mathrm{dist}_{n-1}(s, v) = \lim_{k\to\infty} \mathrm{dist}_k(s, v) = \mathrm{dist}(s, v)$.*
2. *If for all $v \in V$, $\mathrm{dist}_{n-1}(s, v) \leq \mathrm{dist}_n(s, v)$, then there are no negative length cycles.*

Bellman-Ford

Bellman-Ford

(single source) shortest paths:

(single source) shortest paths: source $s \in V$,

Bellman-Ford

(single source) shortest paths: source $s \in V$,
can reach every other node

Bellman-Ford

(single source) shortest paths: source $s \in V$,
can reach every other node

$\texttt{BellmanFord}(G = (V, E),\ \ell : V \to \mathbb{Z},\ s \in V)$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

`BellmanFord(`$G = (V, E)$, $\ell : V \to \mathbb{Z}$, $s \in V$`)`

    for $v \in V$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V,E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n,
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

BellmanFord($G = (V, E)$, $\ell : V \to \mathbb{Z}$, $s \in V$)

for $v \in V$

$\quad d_0[s][v] = \infty$

$d_0[s][s] = 0$

for $1 \le k \le n$, $v \in V$

$\quad d_k[s][v] = d_{k-1}[s][v]$

$\quad$ **for** $u \in N^-(v)$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] =
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v],
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

$\texttt{BellmanFord}(G = (V, E),\ \ell : V \to \mathbb{Z},\ s \in V)$

for $v \in V$

  $d_0[s][v] = \infty$

$d_0[s][s] = 0$

for $1 \le k \le n,\ \ v \in V$

  $d_k[s][v] = d_{k-1}[s][v]$

  for $u \in N^-(v)$

    $d_k[s][v] = \min\{d_k[s][v], d_{k-1}[s][u] +$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
        for u ∈ N^-(v)
            d_k[s][v] = min{d_k[s][v], d_{k-1}[s][u] + ℓ(u, v)}
    for v ∈ V
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N⁻(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
        for u ∈ N^-(v)
            d_k[s][v] = min{d_k[s][v], d_{k-1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n-1}[s][v]
            return ''negative cycle detected''
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ‘‘negative cycle detected’’
    return d_{n−1}[s][·]
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V,E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
        for u ∈ N^-(v)
            d_k[s][v] = min{d_k[s][v], d_{k-1}[s][u] + ℓ(u,v)}
    for v ∈ V
        if d_n[s][v] < d_{n-1}[s][v]
            return ``negative cycle detected''
    return d_{n-1}[s][·]
```

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness:

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ``negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
        for u ∈ N^-(v)
            d_k[s][v] = min{d_k[s][v], d_{k-1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n-1}[s][v]
            return ''negative cycle detected''
    return d_{n-1}[s][·]
```

correctness: clear

complexity:

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N⁻(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ``negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V,E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u,v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ``negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:*

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$,

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ``negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges
- space

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges
- space
  - clearly $O(n^2)$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V,E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k-1}[s][v]
        for u ∈ N^-(v)
            d_k[s][v] = min{d_k[s][v], d_{k-1}[s][u] + ℓ(u,v)}
    for v ∈ V
        if d_n[s][v] < d_{n-1}[s][v]
            return ``negative cycle detected''
    return d_{n-1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges
- space
  - clearly $O(n^2)$
  - *better:*

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges
- space
  - clearly $O(n^2)$
  - *better:* only store $d_{\text{cur}}[s][\cdot]$ and $d_{\text{prev}}[s][\cdot]$

Bellman-Ford

(single source) shortest paths: source $s \in V$, can reach every other node

```
BellmanFord(G = (V, E), ℓ : V → ℤ, s ∈ V)
    for v ∈ V
        d_0[s][v] = ∞
    d_0[s][s] = 0
    for 1 ≤ k ≤ n, v ∈ V
        d_k[s][v] = d_{k−1}[s][v]
        for u ∈ N^−(v)
            d_k[s][v] = min{d_k[s][v], d_{k−1}[s][u] + ℓ(u, v)}
    for v ∈ V
        if d_n[s][v] < d_{n−1}[s][v]
            return ''negative cycle detected''
    return d_{n−1}[s][·]
```

correctness: clear

complexity:

- time
  - clearly $O(n^3)$
  - *better:* $O(mn)$, $d_k[s][\cdot]$ updates along edges
- space
  - clearly $O(n^2)$
  - *better:* only store $d_{\text{cur}}[s][\cdot]$ and $d_{\text{prev}}[s][\cdot] \implies O(n)$

Bellman-Ford (II)

Bellman-Ford (II)

remarks:

Bellman-Ford (II)

remarks:

- compute actual paths

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated,

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

v_{k-1}

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V}$$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s, u) +$$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s, u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from* s in G

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \left\{ \mathrm{dist}_{k-1}(s,u) + \ell(u, v_k) \right\} .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

$\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from* s in G

$\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

$\Longrightarrow$ $O(mn^2)$ time

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \; .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from* s in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

  $\Longrightarrow$ $O(mn^2)$ time
  - *better:*

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' =$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\},$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \left\{ \mathrm{dist}_{k-1}(s,u) + \ell(u, v_k) \right\} .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s in G*

$\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G

$\Longrightarrow$ $O(mn^2)$ time

  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \left\{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\right\} .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \left\{ \text{dist}_{k-1}(s, u) + \ell(u, v_k) \right\} .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\implies$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\implies$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\implies$ *all* negative cycles in G are reachable from s' in G'

  $\implies$ *one* Bellman-Ford required $\implies$ $O(mn)$ time
- directed *acyclic* graphs

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \ .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles
  - can simplify Bellman-Ford so $\text{dist}_k(s, \cdot)$ only updates v_k,

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s,u) + \ell(u, v_k)\} \, .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles
  - can simplify Bellman-Ford so $\mathrm{dist}_k(s,\cdot)$ only updates v_k, according to topological ordering $v_1 \prec v_2 \prec \cdots \prec v_n$

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\mathrm{dist}_{k-1}(s,u) + \ell(u, v_k)\} \; .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s',v)\}_{v \in V})$ with $\ell'(s',v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles
  - can simplify Bellman-Ford so $\mathrm{dist}_k(s,\cdot)$ only updates v_k, according to topological ordering $v_1 \prec v_2 \prec \cdots \prec v_n$ — yields Dijkstra-esque algorithm

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,

$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} .$$

- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles
  - can simplify Bellman-Ford so $\text{dist}_k(s, \cdot)$ only updates v_k, according to topological ordering $v_1 \prec v_2 \prec \cdots \prec v_n$ — yields Dijkstra-esque algorithm

  $\Longrightarrow$ $O(m + n)$ time

Bellman-Ford (II)

remarks:

- compute actual paths by storing pointers indicating *how* $d_k[s][\cdot]$ was updated, e.g.,
$$v_{k-1} = \arg\min_{u \in V} \{\text{dist}_{k-1}(s,u) + \ell(u, v_k)\} \; .$$
- detecting negative cycles
  - Bellman-Ford will detect any negative cycles *reachable from s* in G

  $\Longrightarrow$ one Bellman-Ford call *per vertex* will detect if there is *any* negative cycle in G
  $\Longrightarrow$ $O(mn^2)$ time
  - *better:* consider $G' = (V \cup \{s'\}, E \cup \{(s', v)\}_{v \in V})$ with $\ell'(s', v) = 0$

  $\Longrightarrow$ *all* negative cycles in G are reachable from s' in G'

  $\Longrightarrow$ *one* Bellman-Ford required $\Longrightarrow$ $O(mn)$ time
- directed *acyclic* graphs
  - no (negative) cycles
  - can simplify Bellman-Ford so $\text{dist}_k(s, \cdot)$ only updates v_k, according to topological ordering $v_1 \prec v_2 \prec \cdots \prec v_n$ — yields Dijkstra-esque algorithm

  $\Longrightarrow$ $O(m+n)$ time (**exercise**)

All-Pairs Shortest Paths

All-Pairs Shortest Paths

Definition

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$.

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$,

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$,

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\operatorname{dist}(s, t)$ for all $t \in V$ (single-source)

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
    - non-negative lengths
    - $O((m + n) \log n)$ time

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps),

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps), $O(m + n \log n)$

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps), $O(m + n \log n)$ (Fibonacci heaps)

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps), $O(m + n \log n)$ (Fibonacci heaps)
- *Bellman-Ford:*

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps), $O(m + n \log n)$ (Fibonacci heaps)
- *Bellman-Ford:*
  - arbitrary weights

All-Pairs Shortest Paths

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

single-source:

- *Dijkstra:*
  - non-negative lengths
  - $O((m + n) \log n)$ time (heaps), $O(m + n \log n)$ (Fibonacci heaps)
- *Bellman-Ford:*
  - arbitrary weights
  - $O(mn)$ time

All-Pairs Shortest Paths (II)

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$.

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps),

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)
- n runs of *Bellman-Ford:*

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)
- n runs of *Bellman-Ford:*
  - arbitrary weights

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)
- n runs of *Bellman-Ford:*
  - arbitrary weights
  - $O(n \cdot mn)$ time

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)
- n runs of *Bellman-Ford:*
  - arbitrary weights
  - $O(n \cdot mn)$ time $\mapsto \Theta(n^4)$ if $m = \Theta(n^2)$

All-Pairs Shortest Paths (II)

Definition

$G = (V, E)$ directed (simple) graph, $\ell : E \to \mathbb{Z}$. The **shortest path problem** is to:

- given $s, t \in V$, find a minimum length (s, t)-path
- given $s \in V$, compute $\text{dist}(s, t)$ for all $t \in V$ (single-source)
- compute $\text{dist}(s, t)$ for all $s, t \in V$ (all pairs)

all-pairs:

- n runs of *Dijkstra:*
  - non-negative lengths
  - $O(n \cdot (m + n) \log n)$ time (heaps), $O(n \cdot (m + n \log n))$ (Fibonacci heaps)
- n runs of *Bellman-Ford:*
  - arbitrary weights
  - $O(n \cdot mn)$ time $\mapsto \Theta(n^4)$ if $m = \Theta(n^2)$

question: can we do better?

All-Pairs Shortest Paths (III)

All-Pairs Shortest Paths (III)

idea:

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$.

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$.

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$,

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$.

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$,

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\mathrm{dist}^k(s, t)$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

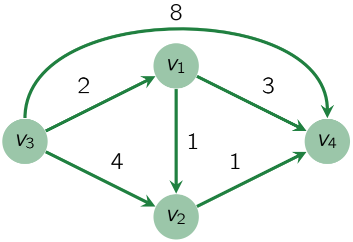
$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\mathrm{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.

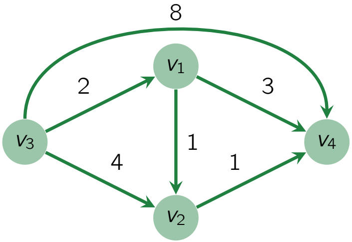


All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\mathrm{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



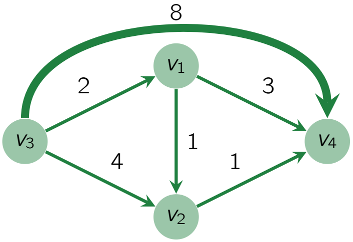
- $\mathrm{dist}^0(v_3, v_4) =$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\mathrm{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



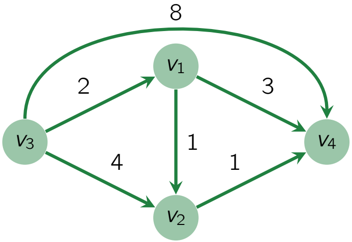
- $\mathrm{dist}^0(v_3, v_4) = \ell(v_3, v_4) = 8$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



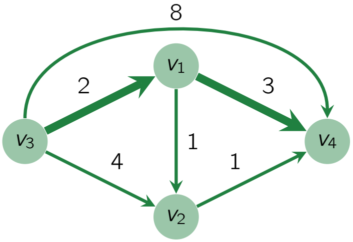
- $\text{dist}^0(v_3, v_4) = \ell(v_3, v_4) = 8$
- $\text{dist}^1(v_3, v_4) =$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



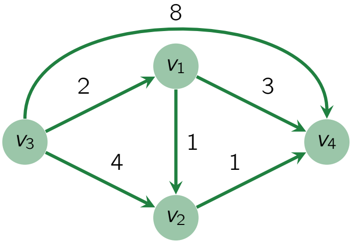
- $\text{dist}^0(v_3, v_4) = \ell(v_3, v_4) = 8$
- $\text{dist}^1(v_3, v_4) = 5$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



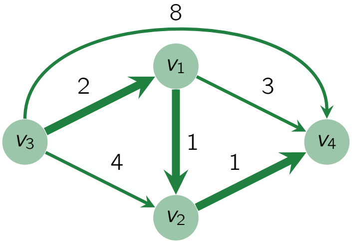
- $\text{dist}^0(v_3, v_4) = \ell(v_3, v_4) = 8$
- $\text{dist}^1(v_3, v_4) = 5$
- $\text{dist}^2(v_3, v_4) =$

All-Pairs Shortest Paths (III)

idea: use a new parameterization of the subproblems

Definition

$G = (V, E)$ directed (simple) graph, with edge length function $\ell : E \to \mathbb{Z}$. Order V as $v_1 \prec v_2 \prec \cdots \prec v_n$. A (u, v)-walk $u = w_0 \to w_1 \to \cdots \to w_i = v$ has **intermediate index** $\leq j$, if $w_1, \ldots, w_{i-1} \in \{v_1, \ldots, v_j\}$. For $s, t \in V$, define $\text{dist}^k(s, t)$ to be the length of the shortest (s, t)-walk of intermediate index $\leq k$.



- $\text{dist}^0(v_3, v_4) = \ell(v_3, v_4) = 8$
- $\text{dist}^1(v_3, v_4) = 5$
- $\text{dist}^2(v_3, v_4) = 4$

All-Pairs Shortest Paths (IV)

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$,

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$,

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with no negative cycles.*

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with no negative cycles. Then for all* $s, t \in V$,
$\text{dist}^0(s, t) = \ell(s, t)$,

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,
$\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) =$$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,
$\text{dist}^0(s, t) = \ell(s, t)$*, and*

$$\text{dist}^k(s, t) = \min \left\{ \right.$$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \end{cases}$$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \end{cases}$$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,
$\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s,v_k) + \text{dist}^{k-1}(v_k,t) \end{cases} .$$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s,t) = \ell(s,t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s,v_k) + \text{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s,v_k) + \text{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

Let $s = w_0$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$*, and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,
$\text{dist}^0(s, t) = \ell(s, t)$*, and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$.

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s,t) = \ell(s,t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s,v_k) + \text{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s,t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s,t)$. There are two cases:

- index $< k$:

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\text{dist}^{k-1}(s, t)$

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\text{dist}^{k-1}(s, t)$
- index $= k$:

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\mathrm{dist}^0(s, t) = \ell(s, t)$*, and*

$$\mathrm{dist}^k(s,t) = \min \begin{cases} \mathrm{dist}^{k-1}(s,t) \\ \mathrm{dist}^{k-1}(s,v_k) + \mathrm{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s,t)-walk of intermediate index $\leq k$ and length $\mathrm{dist}^k(s,t)$. There are two cases:

- index $< k$: hence is of value $\mathrm{dist}^{k-1}(s,t)$
- index $= k$:
  - no negative cycles

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\mathrm{dist}^0(s,t) = \ell(s,t)$, *and*

$$\mathrm{dist}^k(s,t) = \min \begin{cases} \mathrm{dist}^{k-1}(s,t) \\ \mathrm{dist}^{k-1}(s,v_k) + \mathrm{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s,t)-walk of intermediate index $\leq k$ and length $\mathrm{dist}^k(s,t)$. There are two cases:

- index $< k$: hence is of value $\mathrm{dist}^{k-1}(s,t)$
- index $= k$:
  - no negative cycles $\implies$ shortest *walk* is *path*

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\text{dist}^{k-1}(s, t)$
- index $= k$:
  - no negative cycles $\implies$ shortest *walk* is *path* $\implies$ v_k appears exactly once

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$, *and*

$$\text{dist}^k(s, t) = \min \begin{cases} \text{dist}^{k-1}(s, t) \\ \text{dist}^{k-1}(s, v_k) + \text{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\text{dist}^{k-1}(s, t)$
- index $= k$:
  - no negative cycles $\implies$ shortest *walk* is *path* $\implies$ v_k appears exactly once

$\implies$ $s \rightsquigarrow v_k$ path and $v_k \rightsquigarrow t$ path

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$, $\text{dist}^0(s, t) = \ell(s, t)$*, and*

$$\text{dist}^k(s,t) = \min \begin{cases} \text{dist}^{k-1}(s,t) \\ \text{dist}^{k-1}(s,v_k) + \text{dist}^{k-1}(v_k,t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\text{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\text{dist}^{k-1}(s, t)$
- index $= k$:
  - no negative cycles $\implies$ shortest *walk* is *path* $\implies$ v_k appears exactly once

$\implies$ $s \rightsquigarrow v_k$ path and $v_k \rightsquigarrow t$ path are of index $< k$,

All-Pairs Shortest Paths (IV)

Lemma

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with no negative cycles. Then for all $s, t \in V$,
$\mathrm{dist}^0(s, t) = \ell(s, t)$, *and*

$$\mathrm{dist}^k(s, t) = \min \begin{cases} \mathrm{dist}^{k-1}(s, t) \\ \mathrm{dist}^{k-1}(s, v_k) + \mathrm{dist}^{k-1}(v_k, t) \end{cases} .$$

Proof.

Let $s = w_0 \to w_1 \to w_2 \to \cdots \to w_i = t$ be a shortest length (s, t)-walk of intermediate index $\leq k$ and length $\mathrm{dist}^k(s, t)$. There are two cases:

- index $< k$: hence is of value $\mathrm{dist}^{k-1}(s, t)$
- index $= k$:
  - no negative cycles $\implies$ shortest *walk* is *path* $\implies$ v_k appears exactly once

$\implies$ $s \leadsto v_k$ path and $v_k \leadsto t$ path are of index $< k$, and must be *shortest* paths □

Floyd-Warshall

Floyd-Warshall

`FloydWarshall(`$G = (V, E)$`,` $\ell : V \to \mathbb{Z}$`)`

Floyd-Warshall

`FloydWarshall(`$G = (V, E)$`, `$\ell : V \to \mathbb{Z}$`)`

    for $1 \leq i, j \leq n$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i, j ≤ n
        d^0[i][j] = ℓ(i, j)
```

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
```

Floyd-Warshall

$\texttt{FloydWarshall(}G = (V, E)\texttt{, } \ell : V \to \mathbb{Z}\texttt{)}$

for $1 \leq i, j \leq n$

$\quad d^0[i][j] = \ell(i, j)$

for $1 \leq k \leq n$

$\quad$ **for** $1 \leq i, j \leq n$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i, j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i, j ≤ n

            d^k[i][j] =
```

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i, j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i, j ≤ n
            d^k[i][j] = min {
```

Floyd-Warshall

FloydWarshall$(G = (V,E),\ \ell : V \to \mathbb{Z})$

for $1 \le i,j \le n$

$d^0[i][j] = \ell(i,j)$

for $1 \le k \le n$

for $1 \le i,j \le n$

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ \end{cases}$$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] \end{cases}$$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i, j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i, j ≤ n
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

Floyd-Warshall

$\texttt{FloydWarshall}(G = (V, E),\ \ell : V \to \mathbb{Z})$

for $1 \le i, j \le n$

$d^0[i][j] = \ell(i, j)$

for $1 \le k \le n$

for $1 \le i, j \le n$

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

for $1 \le i \le n$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

Floyd-Warshall

FloydWarshall($G = (V, E)$, $\ell : V \to \mathbb{Z}$)

for $1 \leq i, j \leq n$

$d^0[i][j] = \ell(i, j)$

for $1 \leq k \leq n$

for $1 \leq i, j \leq n$

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

for $1 \leq i \leq n$

if $d^n[i][i] < 0$

return ``negative cycle detected''

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i, j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i, j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

remarks:

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ''negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

complexity:

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j] }
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ''negative cycle detected''
    return d^n[·][·]
```

complexity:

- $O(n^3)$ time
- space

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j] }
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ‘‘negative cycle detected’’
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:*

Floyd-Warshall

```
FloydWarshall(G = (V,E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j] }
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ''negative cycle detected''
    return d^n[·][·]
```

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot]$

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k−1}[i][j]
                            { d^{k−1}[i][k] + d^{k−1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i, j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

correctness:

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j] }
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

correctness:

- if *no* negative cycles, correctness is clear

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

correctness:

- if *no* negative cycles, correctness is clear
- if *some* negative cycle, ???

Floyd-Warshall (II)

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$,

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with some negative cycle.*

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.*

Proof.

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.*

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle

k

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, *with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.*

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle

$k = \min_{\text{negative length } C}$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i:v_i \in C} i$.

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C,

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \le n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$.

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k]$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i:v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k]$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C)$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

$\implies$ $d^{k+1}[k][k]$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

$\implies$ $d^{k+1}[k][k] \leq d^k[k][k]$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

$\implies$ $d^{k+1}[k][k] \leq d^k[k][k] < 0$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

$\implies d^{k+1}[k][k] \leq d^k[k][k] < 0$

$\implies d^n[k][k] < 0$

Floyd-Warshall (II)

Proposition

$G = (V, E)$, $\ell : E \to \mathbb{Z}$, with some negative cycle. Then the Floyd-Warshall algorithm correctly detects this cycle.

Proof.

Let $k \leq n$ be the minimum index of a negative length cycle $k = \min_{\text{negative length } C} \max_{i : v_i \in C} i$. Pick such a cycle C, where C is $v_k = w_0 \to w_1 = v_i \to \cdots \to w_j = v_k$. By choice of k,

- $d^{k-1}[k][i] = \text{dist}^{k-1}(k, i) \leq \ell(w_0, w_1)$
- $d^{k-1}[i][k] = \text{dist}^{k-1}(i, k) \leq \ell(w_1, w_2) + \cdots + \ell(w_{j-1}, w_j)$

$\implies$ $d^k[k][k] \leq d^{k-1}[k][i] + d^{k-1}[i][k] = \ell(w_0, w_1) + \cdots + \ell(w_{j-1}, w_j) = \ell(C) < 0$

$\implies$ $d^{k+1}[k][k] \leq d^k[k][k] < 0$

$\implies$ $d^n[k][k] < 0 \implies$ negative cycle detected □

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                              d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ''negative cycle detected''
    return d^n[·][·]
```

$$d^k[i][j] = \min \begin{cases} d^{k-1}[i][j] \\ d^{k-1}[i][k] + d^{k-1}[k][j] \end{cases}$$

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

correctness:

- if *no* negative cycles, correctness is clear
- if *some* negative cycle, ???

Floyd-Warshall

```
FloydWarshall(G = (V, E), ℓ : V → ℤ)
    for 1 ≤ i,j ≤ n
        d^0[i][j] = ℓ(i,j)
    for 1 ≤ k ≤ n
        for 1 ≤ i,j ≤ n
            d^k[i][j] = min { d^{k-1}[i][j]
                            { d^{k-1}[i][k] + d^{k-1}[k][j]
    for 1 ≤ i ≤ n
        if d^n[i][i] < 0
            return ``negative cycle detected''
    return d^n[·][·]
```

remarks:

- compute actual paths by storing pointers indicating *how* $d^k[\cdot][\cdot]$ was updated

complexity:

- $O(n^3)$ time
- space
  - clearly $O(n^3)$
  - *better:* only store $d^{\text{cur}}[\cdot][\cdot]$ and $d^{\text{prev}}[\cdot][\cdot] \implies O(n^2)$

correctness:

- if *no* negative cycles, correctness is clear
- if *some* negative cycle, correctness is now done

Overview (II)

today:

- shortest paths
  - with negative lengths — Bellman-Ford in $O(mn)$ time
  - all-pairs — Floyd-Warshall in $O(n^3)$ time

next lecture:

- *more* dynamic programming

logistics:

- pset2 due R5 — can submit in *groups* of ≤ 3

TOC