

# Kürzeste Wege I

## Algorithmen für verteilte Systeme

Sebastian Forster

Universität Salzburg



Dieses Werk steht unter einer Creative Commons Namensnennung 4.0 International Lizenz.

# Problemstellung

## Single-Source Shortest Paths (SSSP)

**Ziel:** Berechne Distanz  $\text{dist}(s, v)$  für jeden Knoten  $v$  und gegebenen Startknoten  $s$

- Jeder Knoten weiß initial, ob er der Startknoten ist oder nicht
- Am Ende kennt jeder Knoten  $v$  seine Distanz  $\text{dist}(s, v)$

# Problemstellung

## Single-Source Shortest Paths (SSSP)

**Ziel:** Berechne Distanz  $\text{dist}(s, v)$  für jeden Knoten  $v$  und gegebenen Startknoten  $s$

- Jeder Knoten weiß initial, ob er der Startknoten ist oder nicht
- Am Ende kennt jeder Knoten  $v$  seine Distanz  $\text{dist}(s, v)$

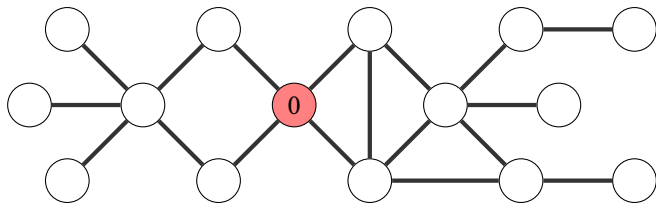
## All-Pairs Shortest Paths (APSP)

**Ziel:** Berechne Distanz  $\text{dist}(u, v)$  für jedes Paar von Knoten  $u$  und  $v$

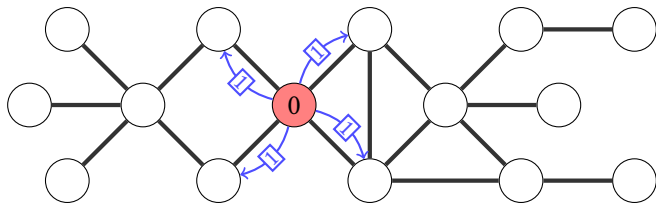
- Am Ende kennt jeder Knoten  $v$  für jeden anderen Knoten  $u$  die Distanz  $\text{dist}(u, v)$

# Ungewichtete Graphen

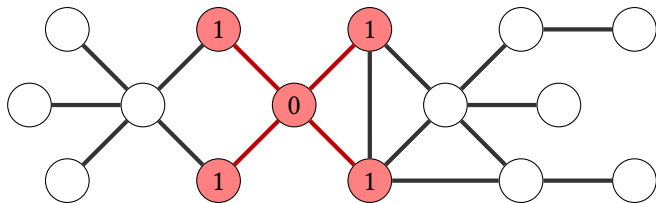
## Wiederholung: Breitensuche



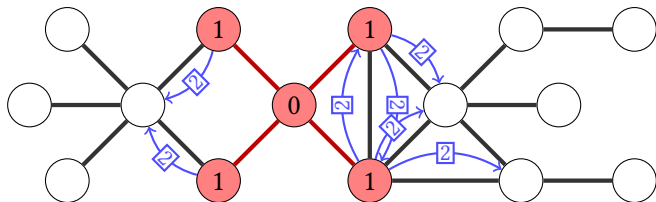
# Wiederholung: Breitensuche



## Wiederholung: Breitensuche

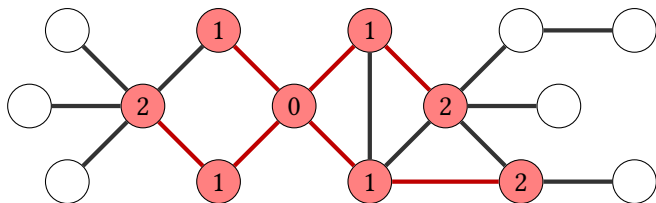


# Wiederholung: Breitensuche

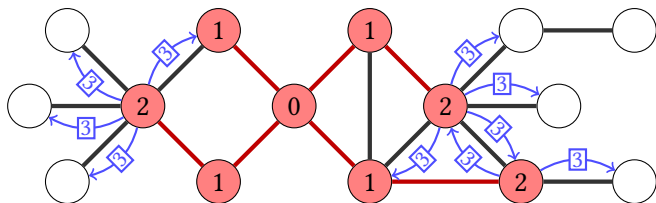




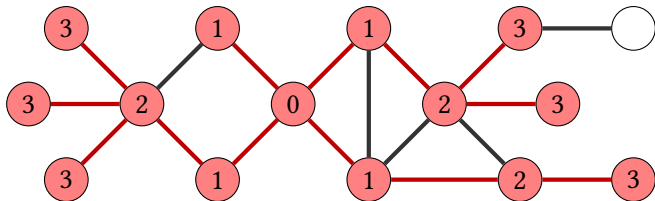
# Wiederholung: Breitensuche



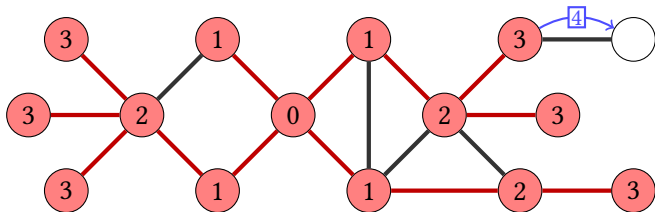
# Wiederholung: Breitensuche



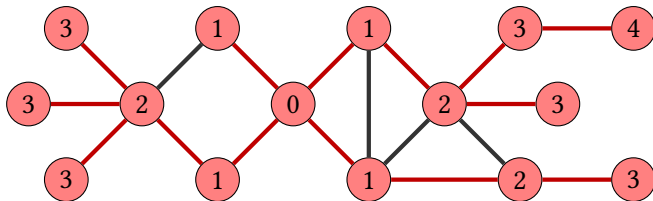
# Wiederholung: Breitensuche



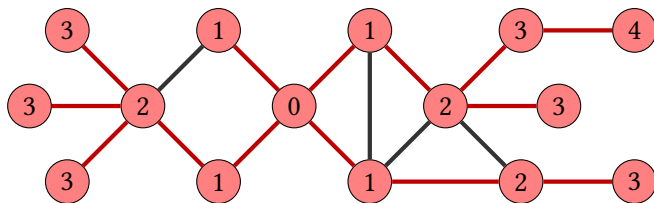
# Wiederholung: Breitensuche



# Wiederholung: Breitensuche



# Wiederholung: Breitensuche



## Theorem

*Im CONGEST Modell kann das SSSP Problem für ungewichtete Graphen in  $O(D)$  Runden gelöst werden.*

# All-Pairs Shortest Paths

**Idee:** Führe Breitensuche für jeden Knoten aus

# All-Pairs Shortest Paths

**Idee:** Führe Breitensuche für jeden Knoten aus

**Naive Umsetzung:**

- Congestion Constraint verhindert parallele Ausführung
- $O(nD)$  Runden



# All-Pairs Shortest Paths

**Idee:** Führe Breitensuche für jeden Knoten aus

**Naive Umsetzung:**

- Congestion Constraint verhindert parallele Ausführung
- $O(nD)$  Runden

**Effizienterer Algorithmus:** Parallelisierung durch Randomisierung

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

### Random Delay Algorithmus:

- Jeder Startknoten  $s_i$  wählt uniform zufälliges  $\delta_i \in \{0, \dots, k - 1\}$

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

### Random Delay Algorithmus:

- Jeder Startknoten  $s_i$  wählt uniform zufälliges  $\delta_i \in \{0, \dots, k - 1\}$
- Knoten  $s_i$  startet Instanz des Algorithmus mit Verzögerung  $\delta_i$

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

### Random Delay Algorithmus:

- Jeder Startknoten  $s_i$  wählt uniform zufälliges  $\delta_i \in \{0, \dots, k - 1\}$
- Knoten  $s_i$  startet Instanz des Algorithmus mit Verzögerung  $\delta_i$

### Lemma

*Der Random Delay Algorithmus berechnet einen Breitensuchbaum für jeden Knoten in  $U$  in  $O(k + D)$  Runden und benötigt Bandbreite  $O(\log^2 n)$  mit hoher Wahrscheinlichkeit.*

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

### Random Delay Algorithmus:

- Jeder Startknoten  $s_i$  wählt uniform zufälliges  $\delta_i \in \{0, \dots, k - 1\}$
- Knoten  $s_i$  startet Instanz des Algorithmus mit Verzögerung  $\delta_i$

### Lemma

*Der Random Delay Algorithmus berechnet einen Breitensuchbaum für jeden Knoten in  $U$  in  $O(k + D)$  Runden und benötigt Bandbreite  $O(\log^2 n)$  mit hoher Wahrscheinlichkeit.*

**Simulation** mit Nachrichtengröße  $O(\log n)$ :  $\Rightarrow O((k + D) \log n)$  Runden

## Parallelisierung durch Random Delay

**Ziel:** Führe Breitensuche für  $k = |U|$  verschiedene Startknoten aus

**Idee:** Nutze aus, dass in Breitensuche jeder Knoten höchstens einmal sendet

### Random Delay Algorithmus:

- Jeder Startknoten  $s_i$  wählt uniform zufälliges  $\delta_i \in \{0, \dots, k-1\}$
- Knoten  $s_i$  startet Instanz des Algorithmus mit Verzögerung  $\delta_i$

### Lemma

*Der Random Delay Algorithmus berechnet einen Breitensuchbaum für jeden Knoten in  $U$  in  $O(k + D)$  Runden und benötigt Bandbreite  $O(\log^2 n)$  mit hoher Wahrscheinlichkeit.*

**Simulation** mit Nachrichtengröße  $O(\log n)$ :  $\Rightarrow O((k + D) \log n)$  Runden

### Theorem

*Im CONGEST Modell gibt es einen Monte-Carlo Algorithmus, der das APSP Problem in  $O(n \log n)$  Runden mit hoher Wahrscheinlichkeit löst.*



# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

### **Beweis:**

$M_{v,i}$ : von Knoten  $v$  in  $i$ -ter Instanz gesendete Nachricht

(Bei Breitensuche gibt es nur eine solche Nachricht)

$r_{v,i}$ : Runde, in der  $M_{v,i}$  ohne Delay gesendet werden würde

# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

### **Beweis:**

$M_{v,i}$ : von Knoten  $v$  in  $i$ -ter Instanz gesendete Nachricht

(Bei Breitensuche gibt es nur eine solche Nachricht)

$r_{v,i}$ : Runde, in der  $M_{v,i}$  ohne Delay gesendet werden würde

**Fixiere Runde  $r$ , Knoten  $v$ , Instanz  $i$ :**

# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

### **Beweis:**

$M_{v,i}$ : von Knoten  $v$  in  $i$ -ter Instanz gesendete Nachricht

(Bei Breitensuche gibt es nur eine solche Nachricht)

$r_{v,i}$ : Runde, in der  $M_{v,i}$  ohne Delay gesendet werden würde

**Fixiere Runde  $r$ , Knoten  $v$ , Instanz  $i$ :**

$$\Pr[M_{v,i} \text{ in Runde } r \text{ von } v \text{ gesendet}] \leq \frac{1}{k}$$

(Entspricht Wahrscheinlichkeit, dass  $r = \delta_i + r_{v,i}$ ; hierbei  $k$  Möglichkeiten für  $\delta_i$ )

# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

### **Beweis:**

$M_{v,i}$ : von Knoten  $v$  in  $i$ -ter Instanz gesendete Nachricht

(Bei Breitensuche gibt es nur eine solche Nachricht)

$r_{v,i}$ : Runde, in der  $M_{v,i}$  ohne Delay gesendet werden würde

**Fixiere Runde  $r$ , Knoten  $v$ , Instanz  $i$ :**

$$\Pr[M_{v,i} \text{ in Runde } r \text{ von } v \text{ gesendet}] \leq \frac{1}{k}$$

(Entspricht Wahrscheinlichkeit, dass  $r = \delta_i + r_{v,i}$ ; hierbei  $k$  Möglichkeiten für  $\delta_i$ )

**Fixiere Runde  $r$ , Knoten  $v$ , Menge von Nachrichten  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$ :**

# Analyse

## Lemma

*In jeder Runde werden mit hoher Wahrscheinlichkeit für jeden Knoten nur Nachrichten aus  $O(\log n)$  Instanzen gleichzeitig gesendet.*

### **Beweis:**

$M_{v,i}$ : von Knoten  $v$  in  $i$ -ter Instanz gesendete Nachricht

(Bei Breitensuche gibt es nur eine solche Nachricht)

$r_{v,i}$ : Runde, in der  $M_{v,i}$  ohne Delay gesendet werden würde

**Fixiere Runde  $r$ , Knoten  $v$ , Instanz  $i$ :**

$$\Pr[M_{v,i} \text{ in Runde } r \text{ von } v \text{ gesendet}] \leq \frac{1}{k}$$

(Entspricht Wahrscheinlichkeit, dass  $r = \delta_i + r_{v,i}$ ; hierbei  $k$  Möglichkeiten für  $\delta_i$ )

**Fixiere Runde  $r$ , Knoten  $v$ , Menge von Nachrichten  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$ :**

$$\Pr[\text{alle Nachrichten aus } \mathcal{M} \text{ in Runde } r \text{ von } v \text{ gesendet}] \leq \left(\frac{1}{k}\right)^{|\mathcal{M}|}$$

(Unabhängigkeit der Ereignisse, da Delays unabhängig gewählt wurden)

## Fortsetzung des Beweises

#Nachrichtmengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right.$

$\left. \text{und alle Nachrichten aus } \mathcal{M} \text{ in Runde } r \text{ von } v \text{ gesendet} \right]$



## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right.$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right.$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right]$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

$$\Pr[\dots] \leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{ek}{\ell}\right)^\ell \left(\frac{1}{k}\right)^\ell$$

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right]$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

$$\Pr[\dots] \leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{ek}{\ell}\right)^\ell \left(\frac{1}{k}\right)^\ell$$

$$= \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{e}{\ell}\right)^\ell$$

## Fortsetzung des Beweises

#Nachrichtensmengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right]$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

$$\begin{aligned} \Pr[\dots] &\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{ek}{\ell}\right)^\ell \left(\frac{1}{k}\right)^\ell \\ &= \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{e}{\ell}\right)^\ell \leq k \left(\frac{1}{2}\right)^{(c+4) \log n} \end{aligned}$$

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right]$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

$$\Pr[\dots] \leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{ek}{\ell}\right)^\ell \left(\frac{1}{k}\right)^\ell$$

$$= \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{e}{\ell}\right)^\ell \leq k \left(\frac{1}{2}\right)^{(c+4) \log n} = k \cdot \frac{1}{n^{c+4}} \leq \frac{1}{n^{c+3}}$$

## Fortsetzung des Beweises

#Nachrichtemengen  $\mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\}$  der Größe  $\ell: \leq \binom{k}{\ell}$

(Eine Nachricht pro Instanz)

$\Pr \left[ \exists \mathcal{M} \subseteq \bigcup_{i=0}^{k-1} \{M_{v,i}\} \text{ so dass } |\mathcal{M}| \geq \max\{(c+4) \log n, 2e\} \text{ (} e = \text{Eulersche Zahl)} \right]$

und alle Nachrichten aus  $\mathcal{M}$  in Runde  $r$  von  $v$  gesendet

$$\leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \binom{k}{\ell} \left(\frac{1}{k}\right)^\ell$$

(Union Bound)

Wegen  $\binom{a}{b} \leq \left(\frac{ea}{b}\right)^b$ :

$$\Pr[\dots] \leq \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{ek}{\ell}\right)^\ell \left(\frac{1}{k}\right)^\ell$$

$$= \sum_{\max\{(c+4) \log n, 2e\} \leq \ell \leq k} \left(\frac{e}{\ell}\right)^\ell \leq k \left(\frac{1}{2}\right)^{(c+4) \log n} = k \cdot \frac{1}{n^{c+4}} \leq \frac{1}{n^{c+3}}$$

$\Rightarrow$  Obere Schranke für alle Knoten und Runden mit Wahrsch.  $1 - 1/n^c$

# Zusammenfassung Random Delay

Wir haben gezeigt:

## Lemma

*Im CONGEST Modell können  $k \leq n$  Instanzen der Breitensuche mit hoher Wahrscheinlichkeit in  $O((k + D) \log n)$  Runden ausgeführt werden.*



# Zusammenfassung Random Delay

Wir haben gezeigt:

## Lemma

*Im CONGEST Modell können  $k \leq n$  Instanzen der Breitensuche mit hoher Wahrscheinlichkeit in  $O((k + D) \log n)$  Runden ausgeführt werden.*

## Verallgemeinerung:

### Lemma

*Sei  $\mathcal{A}$  ein Algorithmus im CONGEST Modell, der  $R(n) = n^{O(1)}$  Runden benötigt und in dem jeder Knoten in insgesamt  $M(n) = n^{O(1)}$  Runden Nachrichten sendet. Dann können  $k = n^{O(1)}$  Instanzen von  $\mathcal{A}$  mit hoher Wahrscheinlichkeit in  $O((R(n) + kM(n)) \log n)$  Runden ausgeführt werden.*

# Gewichtete Graphen

# Modell

## Annahmen:

- Jede Kante  $e = (u, v)$  hat ein Gewicht  $w(u, v)$ , das  $u$  und  $v$  bekannt ist
- Positive, ganzzahlige Kantengewichte von 1 bis  $W$

# Modell

## Annahmen:

- Jede Kante  $e = (u, v)$  hat ein Gewicht  $w(u, v)$ , das  $u$  und  $v$  bekannt ist
- Positive, ganzzahlige Kantengewichte von 1 bis  $W$
- Bandbreite  $O(\log(nW)) = O(\log n + \log W)$
- Somit: Summe von bis zu  $n$  Kantengewichten kann in einer Nachricht gesendet werden

# Modell

## Annahmen:

- Jede Kante  $e = (u, v)$  hat ein Gewicht  $w(u, v)$ , das  $u$  und  $v$  bekannt ist
- Positive, ganzzahlige Kantengewichte von 1 bis  $W$
- Bandbreite  $O(\log(nW)) = O(\log n + \log W)$
- Somit: Summe von bis zu  $n$  Kantengewichten kann in einer Nachricht gesendet werden

## Achtung

Kantengewichte repräsentieren Kosten, keine Delays. Direkte Kommunikation mit Nachbarn dauert nur eine Runde.

# Modell

## Annahmen:

- Jede Kante  $e = (u, v)$  hat ein Gewicht  $w(u, v)$ , das  $u$  und  $v$  bekannt ist
- Positive, ganzzahlige Kantengewichte von 1 bis  $W$
- Bandbreite  $O(\log(nW)) = O(\log n + \log W)$
- Somit: Summe von bis zu  $n$  Kantengewichten kann in einer Nachricht gesendet werden

## Achtung

Kantengewichte repräsentieren Kosten, keine Delays. Direkte Kommunikation mit Nachbarn dauert nur eine Runde.

Weiterhin bezeichnet  $D$  den Durchmesser des *ungewichteten* Netzwerks.

# Modell

## Annahmen:

- Jede Kante  $e = (u, v)$  hat ein Gewicht  $w(u, v)$ , das  $u$  und  $v$  bekannt ist
- Positive, ganzzahlige Kantengewichte von 1 bis  $W$
- Bandbreite  $O(\log(nW)) = O(\log n + \log W)$
- Somit: Summe von bis zu  $n$  Kantengewichten kann in einer Nachricht gesendet werden

## Achtung

Kantengewichte repräsentieren Kosten, keine Delays. Direkte Kommunikation mit Nachbarn dauert nur eine Runde.

Weiterhin bezeichnet  $D$  den Durchmesser des *ungewichteten* Netzwerks.

## Notation

$\text{dist}^h(s, v)$  = Länge des kürzesten Wegs von  $s$  nach  $v$  mit höchstens  $h$  Kanten

# Bellman-Ford Algorithmus

Initialisierung in Runde 1:

$$\delta_1(v) = \begin{cases} 0 & \text{falls } v = s \\ \infty & \text{andernfalls} \end{cases}$$



# Bellman-Ford Algorithmus

Initialisierung in Runde 1:

$$\delta_1(v) = \begin{cases} 0 & \text{falls } v = s \\ \infty & \text{andernfalls} \end{cases}$$

Update in Runde  $r \geq 2$ :

$$\delta_r(v) = \min_{(u,v) \in E} (\delta_{r-1}(u) + w(u,v))$$

# Bellman-Ford Algorithmus

Initialisierung in Runde 1:

$$\delta_1(v) = \begin{cases} 0 & \text{falls } v = s \\ \infty & \text{andernfalls} \end{cases}$$

Update in Runde  $r \geq 2$ :

$$\delta_r(v) = \min_{(u,v) \in E} (\delta_{r-1}(u) + w(u,v))$$

## Invariante

Nach Runde  $r$  ist  $\delta_r(v) = \text{dist}^{r-1}(s, v)$  (entspricht also der Länge des kürzesten Wegs von  $s$  nach  $v$  mit höchstens  $r - 1$  Kanten).

# Bellman-Ford Algorithmus

Initialisierung in Runde 1:

$$\delta_1(v) = \begin{cases} 0 & \text{falls } v = s \\ \infty & \text{andernfalls} \end{cases}$$

Update in Runde  $r \geq 2$ :

$$\delta_r(v) = \min_{(u,v) \in E} (\delta_{r-1}(u) + w(u,v))$$

## Invariante

Nach Runde  $r$  ist  $\delta_r(v) = \text{dist}^{r-1}(s, v)$  (entspricht also der Länge des kürzesten Wegs von  $s$  nach  $v$  mit höchstens  $r - 1$  Kanten).

**Korrektheit:** Absolut kürzester Weg hat höchstens  $n - 1$  Kanten

# Bellman-Ford Algorithmus

Initialisierung in Runde 1:

$$\delta_1(v) = \begin{cases} 0 & \text{falls } v = s \\ \infty & \text{andernfalls} \end{cases}$$

Update in Runde  $r \geq 2$ :

$$\delta_r(v) = \min_{(u,v) \in E} (\delta_{r-1}(u) + w(u,v))$$

## Invariante

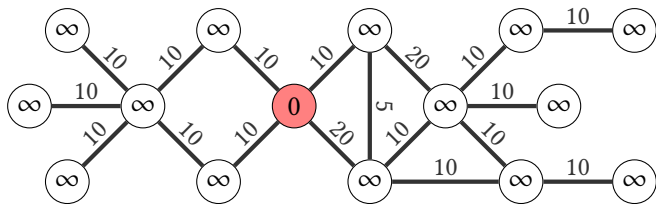
Nach Runde  $r$  ist  $\delta_r(v) = \text{dist}^{r-1}(s, v)$  (entspricht also der Länge des kürzesten Wegs von  $s$  nach  $v$  mit höchstens  $r - 1$  Kanten).

**Korrektheit:** Absolut kürzester Weg hat höchstens  $n - 1$  Kanten

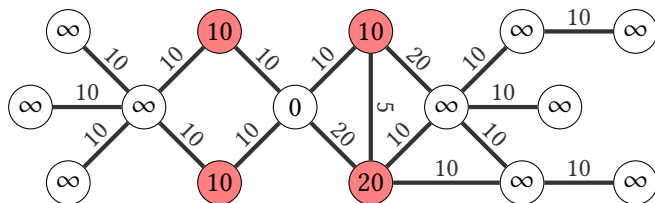
## Theorem

*Im CONGEST Modell kann das SSSP Problem für gewichtete Graphen in  $O(n)$  Runden gelöst werden.*

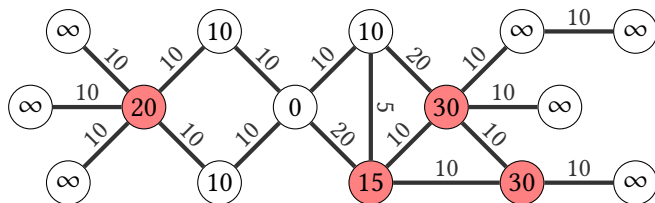
# Beispiel



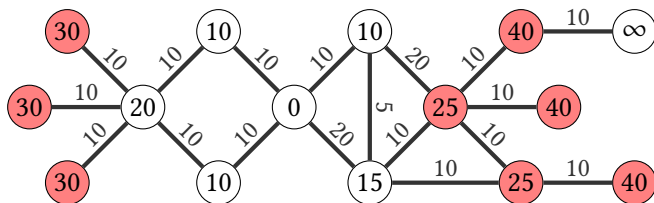
# Beispiel



# Beispiel

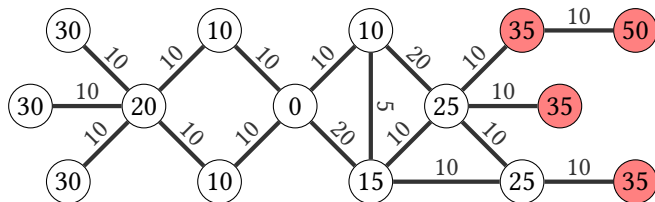


# Beispiel

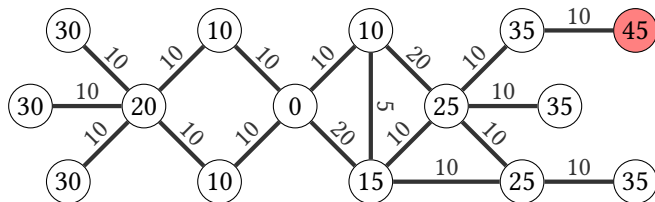




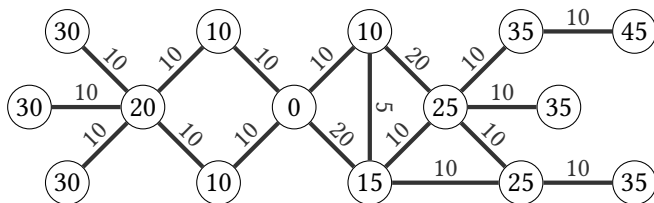
# Beispiel



# Beispiel



# Beispiel



# All-Pairs Shortest Paths

**Idee:** Führe SSSP-Algorithmus für jeden Knoten aus

# All-Pairs Shortest Paths

**Idee:** Führe SSSP-Algorithmus für jeden Knoten aus

**Umsetzung mit Random-Delay Technik:**

- Im Bellman-Ford Algorithmus sendet jeder Knoten in jeder Runde

# All-Pairs Shortest Paths

**Idee:** Führe SSSP-Algorithmus für jeden Knoten aus

**Umsetzung mit Random-Delay Technik:**

- Im Bellman-Ford Algorithmus sendet jeder Knoten in jeder Runde
- Somit: Laufzeit  $O(n^2 \log n)$

# All-Pairs Shortest Paths

**Idee:** Führe SSSP-Algorithmus für jeden Knoten aus

**Umsetzung mit Random-Delay Technik:**

- Im Bellman-Ford Algorithmus sendet jeder Knoten in jeder Runde
- Somit: Laufzeit  $O(n^2 \log n)$
- Nicht besser als naive sequentielle Ausführung mit Laufzeit  $O(n^2)$

# All-Pairs Shortest Paths

**Idee:** Führe SSSP-Algorithmus für jeden Knoten aus

**Umsetzung mit Random-Delay Technik:**

- Im Bellman-Ford Algorithmus sendet jeder Knoten in jeder Runde
- Somit: Laufzeit  $O(n^2 \log n)$
- Nicht besser als naive sequentielle Ausführung mit Laufzeit  $O(n^2)$

**Ziel:** SSSP mit geringerer Anzahl an Nachrichten pro Knoten



## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

**Algorithmus:** Sobald für Knoten  $u \neq s$  die erste Nachricht mit Distanz  $\delta$  empfangen wurde oder Knoten  $u = s$  mit Distanz 0 initialisiert wurde:

- Sei  $r$  die aktuelle Runde
- Für jeden Nachbarknoten  $v$ : Sende Nachricht mit Distanz  $\delta + w(u, v)$  in Runde  $r + w(u, v) - 1$  (Empfang durch  $v$  in Runde  $r + w(u, v)$ )

## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

**Algorithmus:** Sobald für Knoten  $u \neq s$  die erste Nachricht mit Distanz  $\delta$  empfangen wurde oder Knoten  $u = s$  mit Distanz 0 initialisiert wurde:

- Sei  $r$  die aktuelle Runde
- Für jeden Nachbarknoten  $v$ : Sende Nachricht mit Distanz  $\delta + w(u, v)$  in Runde  $r + w(u, v) - 1$  (Empfang durch  $v$  in Runde  $r + w(u, v)$ )

### Invariante

Für jeden Knoten  $v$  gilt: Die erste Distanznachricht wird in Runde  $\text{dist}(s, v) + 1$  empfangen; sie hat den Inhalt  $\text{dist}(s, v)$ .

## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

**Algorithmus:** Sobald für Knoten  $u \neq s$  die erste Nachricht mit Distanz  $\delta$  empfangen wurde oder Knoten  $u = s$  mit Distanz 0 initialisiert wurde:

- Sei  $r$  die aktuelle Runde
- Für jeden Nachbarknoten  $v$ : Sende Nachricht mit Distanz  $\delta + w(u, v)$  in Runde  $r + w(u, v) - 1$  (Empfang durch  $v$  in Runde  $r + w(u, v)$ )

### Invariante

Für jeden Knoten  $v$  gilt: Die erste Distanznachricht wird in Runde  $\text{dist}(s, v) + 1$  empfangen; sie hat den Inhalt  $\text{dist}(s, v)$ .

### Lemma

*In  $T + 1$  Runden wird  $\text{dist}(s, v)$  für jeden Knoten  $v$  mit  $\text{dist}(s, v) \leq T$  berechnet, wobei jeder Knoten höchstens einmal Nachrichten sendet (für alle  $T \leq nW$ ).*

## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

**Algorithmus:** Sobald für Knoten  $u \neq s$  die erste Nachricht mit Distanz  $\delta$  empfangen wurde oder Knoten  $u = s$  mit Distanz 0 initialisiert wurde:

- Sei  $r$  die aktuelle Runde
- Für jeden Nachbarknoten  $v$ : Sende Nachricht mit Distanz  $\delta + w(u, v)$  in Runde  $r + w(u, v) - 1$  (Empfang durch  $v$  in Runde  $r + w(u, v)$ )

### Invariante

Für jeden Knoten  $v$  gilt: Die erste Distanznachricht wird in Runde  $\text{dist}(s, v) + 1$  empfangen; sie hat den Inhalt  $\text{dist}(s, v)$ .

### Lemma

*In  $T + 1$  Runden wird  $\text{dist}(s, v)$  für jeden Knoten  $v$  mit  $\text{dist}(s, v) \leq T$  berechnet, wobei jeder Knoten höchstens einmal Nachrichten sendet (für alle  $T \leq nW$ ).*

Zur Berechnung von  $\text{dist}(s, v)$  sind  $O(nW)$  Runden ausreichend

## Gewichtete Breitensuche

**Idee:** Simuliere gewichteten Graph als ungewichteten Graph, indem jede Kante durch einen Pfad entsprechender Länge ersetzt wird

**Algorithmus:** Sobald für Knoten  $u \neq s$  die erste Nachricht mit Distanz  $\delta$  empfangen wurde oder Knoten  $u = s$  mit Distanz 0 initialisiert wurde:

- Sei  $r$  die aktuelle Runde
- Für jeden Nachbarknoten  $v$ : Sende Nachricht mit Distanz  $\delta + w(u, v)$  in Runde  $r + w(u, v) - 1$  (Empfang durch  $v$  in Runde  $r + w(u, v)$ )

### Invariante

Für jeden Knoten  $v$  gilt: Die erste Distanznachricht wird in Runde  $\text{dist}(s, v) + 1$  empfangen; sie hat den Inhalt  $\text{dist}(s, v)$ .

### Lemma

*In  $T + 1$  Runden wird  $\text{dist}(s, v)$  für jeden Knoten  $v$  mit  $\text{dist}(s, v) \leq T$  berechnet, wobei jeder Knoten höchstens einmal Nachrichten sendet (für alle  $T \leq nW$ ).*

Zur Berechnung von  $\text{dist}(s, v)$  sind  $O(nW)$  Runden ausreichend

→ „Pseudopolynomieller“ Algorithmus

# All-Pairs Shortest Paths

Mit gewichteter Breitensuche:

## Lemma

*Im CONGEST Modell kann das SSSP Problem für gewichtete Graphen in  $O(nW)$  Runden gelöst werden, wobei jeder Knoten höchstens einmal Nachrichten sendet.*

# All-Pairs Shortest Paths

Mit gewichteter Breitensuche:

## Lemma

*Im CONGEST Modell kann das SSSP Problem für gewichtete Graphen in  $O(nW)$  Runden gelöst werden, wobei jeder Knoten höchstens einmal Nachrichten sendet.*

Erinnerung:

## Lemma

*Sei  $\mathcal{A}$  ein Algorithmus im CONGEST Modell, der  $R(n) = n^{O(1)}$  Runden benötigt und in dem jeder Knoten in insgesamt  $M(n) = n^{O(1)}$  Runden Nachrichten sendet. Dann können  $k = n^{O(1)}$  Instanzen von  $\mathcal{A}$  mit hoher Wahrscheinlichkeit in  $O((R(n) + kM(n)) \log n)$  Runden ausgeführt werden.*



# All-Pairs Shortest Paths

Mit gewichteter Breitensuche:

## Lemma

*Im CONGEST Modell kann das SSSP Problem für gewichtete Graphen in  $O(nW)$  Runden gelöst werden, wobei jeder Knoten höchstens einmal Nachrichten sendet.*

Erinnerung:

## Lemma

*Sei  $\mathcal{A}$  ein Algorithmus im CONGEST Modell, der  $R(n) = n^{O(1)}$  Runden benötigt und in dem jeder Knoten in insgesamt  $M(n) = n^{O(1)}$  Runden Nachrichten sendet. Dann können  $k = n^{O(1)}$  Instanzen von  $\mathcal{A}$  mit hoher Wahrscheinlichkeit in  $O((R(n) + kM(n)) \log n)$  Runden ausgeführt werden.*

Somit folgt:

## Theorem

*Im CONGEST Modell gibt es einen Monte-Carlo Algorithmus, der das APSP Problem für gewichtete Graphen in  $O(nW \log n)$  Runden mit hoher Wahrscheinlichkeit löst.*

# Zusammenfassung

- SSSP für ungewichtete Graphen:  $O(D)$  Runden  
Breitensuche
- APSP für ungewichtete Graphen:  $O(n \log n)$  Runden  
Parallelisierung von Breitensuche mit Random Delay Technik
- SSSP für gewichtete Graphen:  $O(n)$  Runden  
Bellman-Ford Algorithmus
- APSP für gewichtete Graphen:  $O(nW \log n)$  Runden  
Parallelisierung von gewichteter Breitensuche mit Random Delay Technik

## Literatur:

- Danupon Nanongkai. „Distributed Approximation Algorithms for Weighted Shortest Paths“. In: *Proc. of the Symposium on Theory of Computing (STOC)*. 2014, S. 565–573
- Frank Thomson Leighton, Bruce M. Maggs, Satish Rao. „Packet Routing and Job-Shop Scheduling in  $O(\text{Congestion} + \text{Dilation})$  Steps“. *Combinatorica* 14(2): 167–186 (1994)