Einführung CONGEST Modell

Algorithmen für verteilte Systeme

Sebastian Forster

Universität Salzburg

Congestion

Modell

- Modellierung des Netzwerks als ungerichteter, zusammenhängender Graph $G = (V, E)$
  - Knoten V modellieren Prozessoren, Kanten E modellieren (symmetrische) Kommunikationsverbindungen
  - $n = |V|$, $m = |E|$
- Rundenbasierte Kommunikation:
  - Synchronisation durch globale Uhr
  - In jeder Runde darf jeder Knoten pro Nachbar eine Nachricht versenden
  - Rundenablauf: Empfangen (entfällt in Runde 1) – Berechnen – Senden
- Knoten haben eindeutige ID, die ihnen und ihren Nachbarn bekannt sind
- Non-uniform: n ist globales Wissen

Modell

- Modellierung des Netzwerks als ungerichteter, zusammenhängender Graph $G = (V, E)$
  - Knoten V modellieren Prozessoren, Kanten E modellieren (symmetrische) Kommunikationsverbindungen
  - $n = |V|$, $m = |E|$
- Rundenbasierte Kommunikation:
  - Synchronisation durch globale Uhr
  - In jeder Runde darf jeder Knoten pro Nachbar eine Nachricht versenden
  - Rundenablauf: Empfangen (entfällt in Runde 1) – Berechnen – Senden
- Knoten haben eindeutige ID, die ihnen und ihren Nachbarn bekannt sind
- Non-uniform: n ist globales Wissen

CONGEST Modell:

- Nachrichtengröße: $O(\log n)$ Bits
  Bandbreitenbeschränkung
- Länge von IDs: $O(\log n)$ Bits

Modell

- Modellierung des Netzwerks als ungerichteter, zusammenhängender Graph $G = (V, E)$
  - Knoten V modellieren Prozessoren, Kanten E modellieren (symmetrische) Kommunikationsverbindungen
  - $n = |V|$, $m = |E|$
- Rundenbasierte Kommunikation:
  - Synchronisation durch globale Uhr
  - In jeder Runde darf jeder Knoten pro Nachbar eine Nachricht versenden
  - Rundenablauf: Empfangen (entfällt in Runde 1) – Berechnen – Senden
- Knoten haben eindeutige ID, die ihnen und ihren Nachbarn bekannt sind
- Non-uniform: n ist globales Wissen

CONGEST Modell:

- Nachrichtengröße: $O(\log n)$ Bits
  Bandbreitenbeschränkung
- Länge von IDs: $O(\log n)$ Bits

LOCAL Modell:

- Nachrichtengröße: unbegrenzt
  Keine Bandbreitenbeschränkung
- Länge von IDs: unbegrenzt

Plan für heute

Algorithmische Techniken für das CONGEST Modell:

1. Broadcast
2. Breitensuchbaum
3. Aggregation
4. Pipelining
5. Queuing

Broadcast

Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

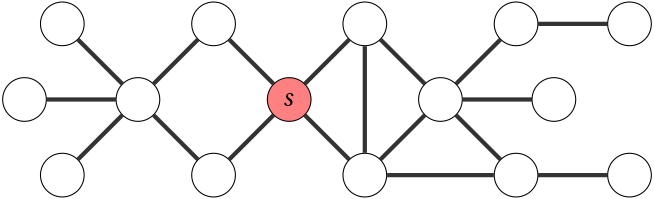
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

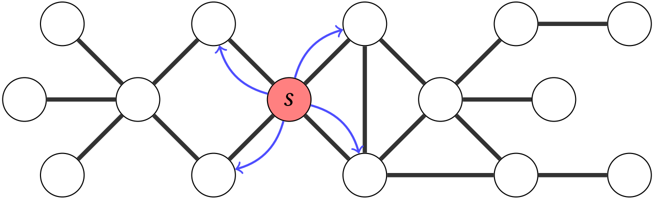


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

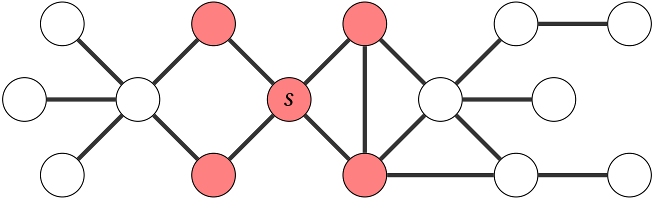


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

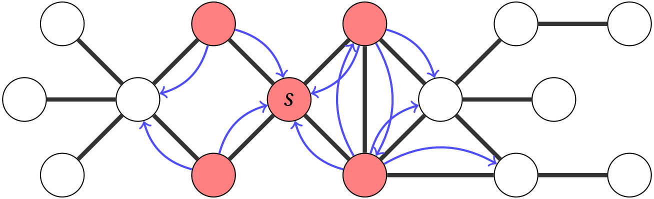


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

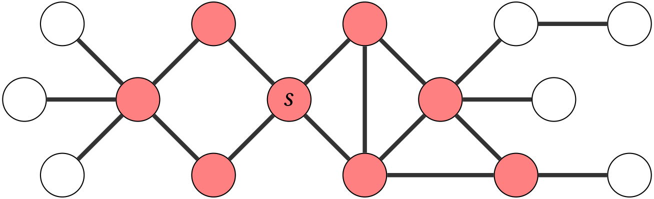


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

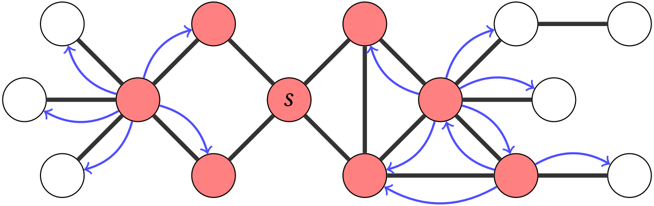


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

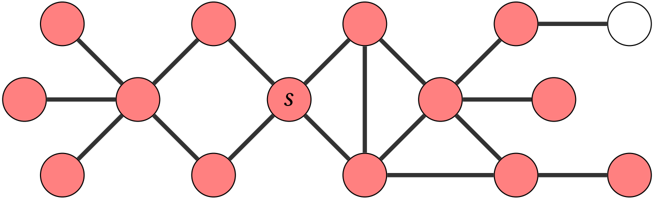


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

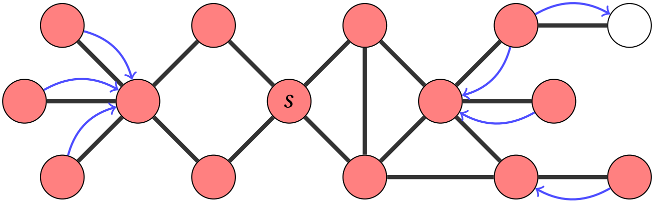


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

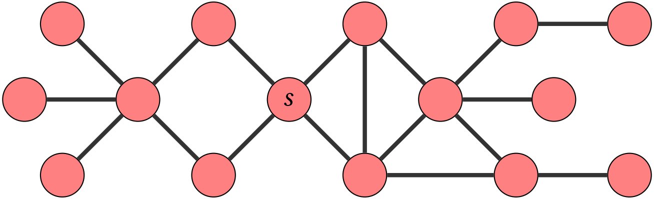


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

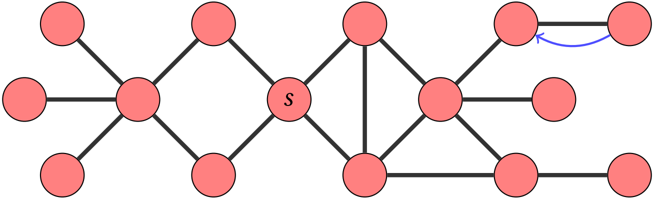


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

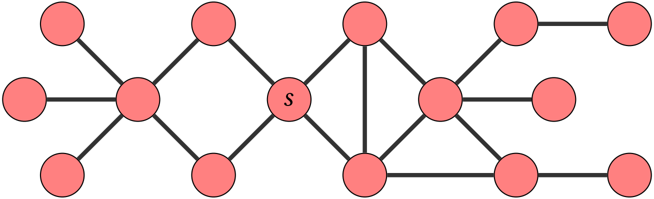


Broadcast-Algorithmus

Ausgangssituation: Knoten s hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\mathrm{dist}(s, v) \geq 1$

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\mathrm{dist}(s, v) \geq 1$

- **Basisfall** ($\mathrm{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \mathrm{dist}(s, v) + 1$

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\mathrm{dist}(s, v) \geq 1$

- **Basisfall** ($\mathrm{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \mathrm{dist}(s, v) + 1$
- **Induktionsschritt** ($\mathrm{dist}(s, v) \geq 2$):

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\operatorname{dist}(s, v) + 1$ die Information I.

$\operatorname{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\operatorname{dist}(s, v) \geq 1$

- **Basisfall** ($\operatorname{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \operatorname{dist}(s, v) + 1$
- **Induktionsschritt** ($\operatorname{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\operatorname{dist}(s, v) + 1$ die Information I.

$\operatorname{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\operatorname{dist}(s, v) \geq 1$

- **Basisfall** ($\operatorname{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \operatorname{dist}(s, v) + 1$
- **Induktionsschritt** ($\operatorname{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\operatorname{dist}(s, u) = \operatorname{dist}(s, v) - 1$

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\mathrm{dist}(s, v) \geq 1$

- **Basisfall** ($\mathrm{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \mathrm{dist}(s, v) + 1$
- **Induktionsschritt** ($\mathrm{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\mathrm{dist}(s, u) = \mathrm{dist}(s, v) - 1$
  - Nach IH: u hat Information I spätestens in Runde $\mathrm{dist}(s, u) + 1$ empfangen und an alle Nachbarn gesendet

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\operatorname{dist}(s, v) + 1$ die Information I.

$\operatorname{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\operatorname{dist}(s, v) \geq 1$

- **Basisfall** ($\operatorname{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \operatorname{dist}(s, v) + 1$
- **Induktionsschritt** ($\operatorname{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\operatorname{dist}(s, u) = \operatorname{dist}(s, v) - 1$
  - Nach IH: u hat Information I spätestens in Runde $\operatorname{dist}(s, u) + 1$ empfangen und an alle Nachbarn gesendet
  - Da v Nachbar von u ist, empfängt v Information I spätestens in Runde $\operatorname{dist}(s, u) + 2 = \operatorname{dist}(s, v) + 1$

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\text{dist}(s, v) + 1$ die Information I.

$\text{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\text{dist}(s, v) \geq 1$

- **Basisfall** ($\text{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \text{dist}(s, v) + 1$
- **Induktionsschritt** ($\text{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\text{dist}(s, u) = \text{dist}(s, v) - 1$
  - Nach IH: u hat Information I spätestens in Runde $\text{dist}(s, u) + 1$ empfangen und an alle Nachbarn gesendet
  - Da v Nachbar von u ist, empfängt v Information I spätestens in Runde $\text{dist}(s, u) + 2 = \text{dist}(s, v) + 1$
- **Korrektheit**:

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\mathrm{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\mathrm{dist}(s, v) \geq 1$

- **Basisfall** ($\mathrm{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \mathrm{dist}(s, v) + 1$
- **Induktionsschritt** ($\mathrm{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\mathrm{dist}(s, u) = \mathrm{dist}(s, v) - 1$
  - Nach IH: u hat Information I spätestens in Runde $\mathrm{dist}(s, u) + 1$ empfangen und an alle Nachbarn gesendet
  - Da v Nachbar von u ist, empfängt v Information I spätestens in Runde $\mathrm{dist}(s, u) + 2 = \mathrm{dist}(s, v) + 1$
- **Korrektheit**:
  - Da Netzwerk zusammenhängend, ist $\mathrm{dist}(s, v)$ für jeden Knoten v *endlich*

Korrektheit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\text{dist}(s, v) + 1$ die Information I.

$\text{dist}(s, v)$ (*Distanz* von s nach v:) Länge des kürzesten Wegs von s nach v

Beobachtung: Für $v \neq s$ gilt $\text{dist}(s, v) \geq 1$

- **Basisfall** ($\text{dist}(s, v) = 1$): v empfängt I von s in Runde $2 = \text{dist}(s, v) + 1$
- **Induktionsschritt** ($\text{dist}(s, v) \geq 2$):
  - Sei u Vorgängerknoten von v auf kürzestem Weg von s nach v
  - Es gilt: $\text{dist}(s, u) = \text{dist}(s, v) - 1$
  - Nach IH: u hat Information I spätestens in Runde $\text{dist}(s, u) + 1$ empfangen und an alle Nachbarn gesendet
  - Da v Nachbar von u ist, empfängt v Information I spätestens in Runde $\text{dist}(s, u) + 2 = \text{dist}(s, v) + 1$
- **Korrektheit**:
  - Da Netzwerk zusammenhängend, ist $\text{dist}(s, v)$ für jeden Knoten v *endlich*
  - Somit: Nach endlicher Rundenzahl kennt jeder Knoten Information I

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \mathrm{dist}(s, v) + 1$

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \mathrm{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \mathrm{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist $\mathrm{Ecc}(u) = \max_v \mathrm{dist}(u, v)$.

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\text{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \text{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist $\text{Ecc}(u) = \max_v \text{dist}(u, v)$.

Definition

Der **Durchmesser** des Netzwerks ist $D = \max_{u,v} \text{dist}(u, v)$.

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\operatorname{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \operatorname{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist $\operatorname{Ecc}(u) = \max_v \operatorname{dist}(u, v)$.

Definition

Der **Durchmesser** des Netzwerks ist $D = \max_{u,v} \operatorname{dist}(u, v)$.

$D = \max_u \operatorname{Ecc}(u)$

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\text{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \text{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist $\text{Ecc}(u) = \max_v \text{dist}(u, v)$.

Definition

Der **Durchmesser** des Netzwerks ist $D = \max_{u,v} \text{dist}(u, v)$.

$D = \max_u \text{Ecc}(u)$, daher: $\text{Ecc}(s) \leq D$

Laufzeit

Induktionshypothese

Für jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + 1$ die Information I.

$\Rightarrow$ Anzahl an Runden: $\max_v \mathrm{dist}(s, v) + 1$

Definition

Die **Exzentrität** eines Knotens u ist $\mathrm{Ecc}(u) = \max_v \mathrm{dist}(u, v)$.

Definition

Der **Durchmesser** des Netzwerks ist $D = \max_{u,v} \mathrm{dist}(u, v)$.

$D = \max_u \mathrm{Ecc}(u)$, daher: $\mathrm{Ecc}(s) \leq D$

Laufzeit Broadcast-Algorithmus: $O(\mathrm{Ecc}(s)) = O(D) = O(n)$ Runden

Nachrichtenkomplexität

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Nachrichtenkomplexität

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Analyse:

- Jeder Knoten leitet Information I *einmal* an alle Nachbarn weiter

Nachrichtenkomplexität

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Analyse:

- Jeder Knoten leitet Information I *einmal* an alle Nachbarn weiter
- Grad(v) Nachrichten pro Knoten v

Nachrichtenkomplexität

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Analyse:

- Jeder Knoten leitet Information I *einmal* an alle Nachbarn weiter
- $\mathrm{Grad}(v)$ Nachrichten pro Knoten v
- Insgesamt: $\sum_v \mathrm{Grad}(v) = 2m = O(m)$ Nachrichten

Breitensuchbaum

Spannbaum durch Breitensuche

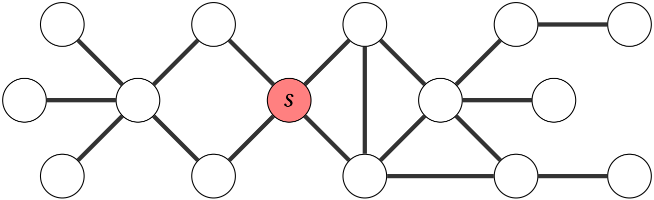
Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn

Spannbaum durch Breitensuche

Algorithmus:

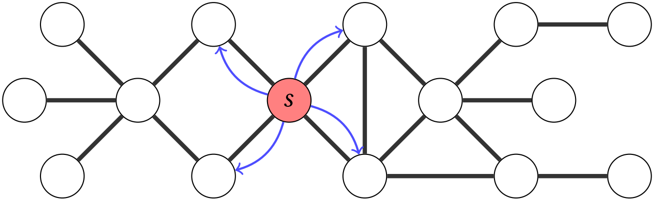
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten
    (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

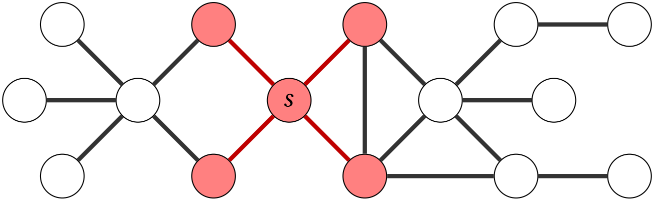
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten
    (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind“-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

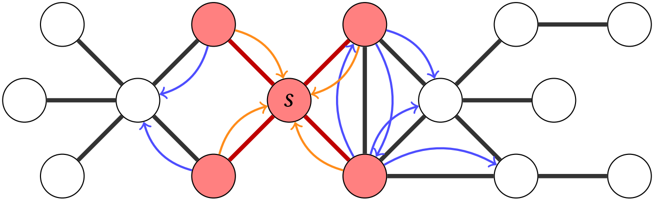
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

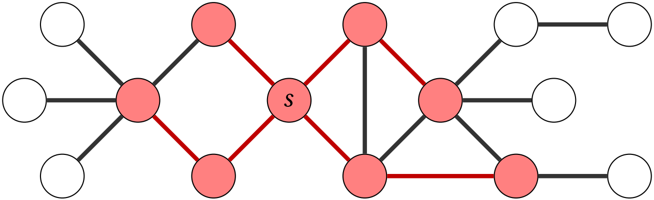
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten
    (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

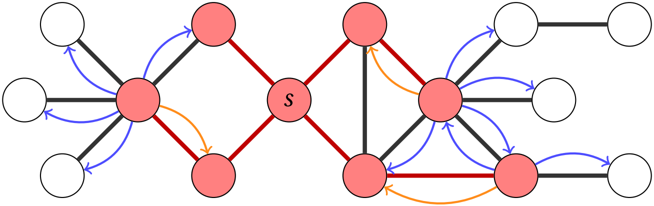
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

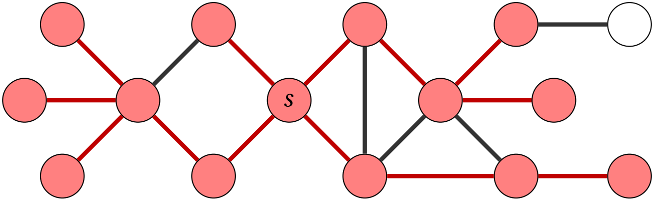
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten
    (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind“-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

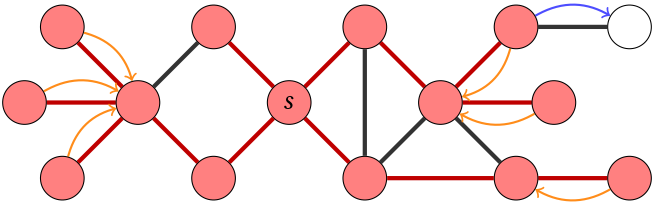
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind“-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

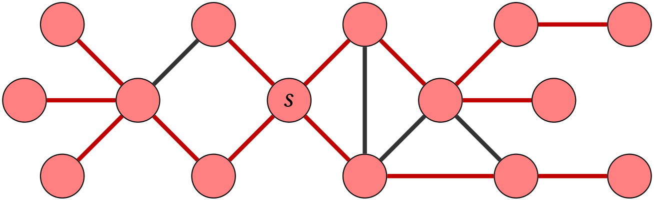
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

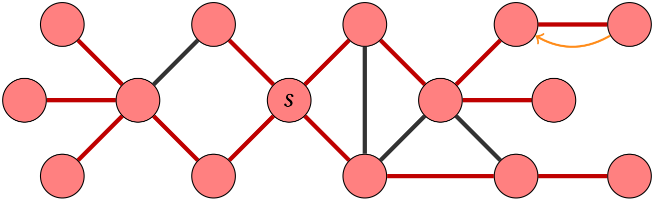
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

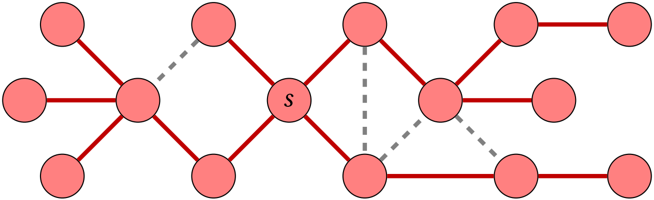
- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind“-Nachricht an Elternknoten und I an alle anderen Nachbarn



Spannbaum durch Breitensuche

Algorithmus:

- s sendet in Runde 1 Information I an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ Information I das erste Mal empfängt:
  - v speichert *einen* der Sender von I als Elternknoten (Tie-Breaking: Zum Beispiel, Knoten mit kleinerer ID wird bevorzugt)
  - v sendet „Kind"-Nachricht an Elternknoten und I an alle anderen Nachbarn



Impliziter Spannbaum: Jeder Knoten kennt Elternknoten und Kinder

Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn

Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten

Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

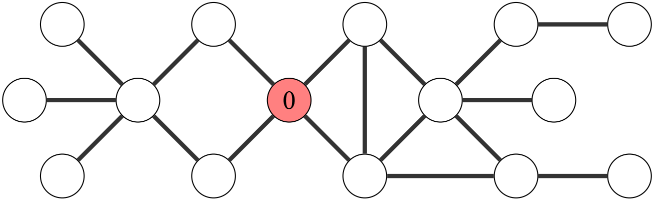
- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
    - v speichert einen der Sender von d als Elternknoten
    - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

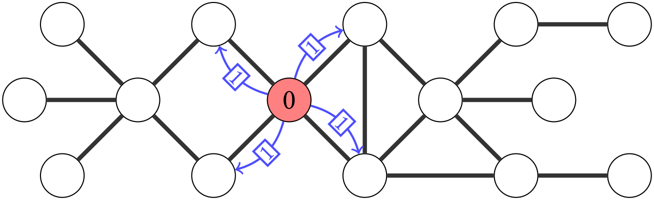


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

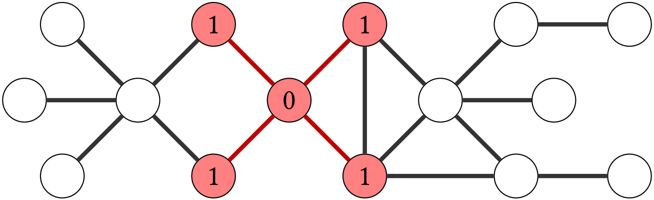


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

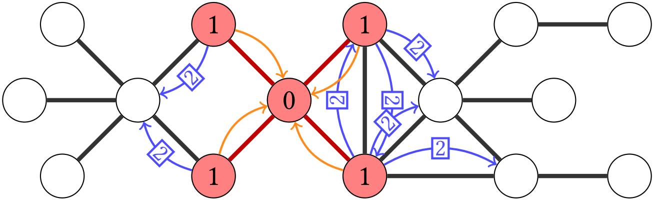


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

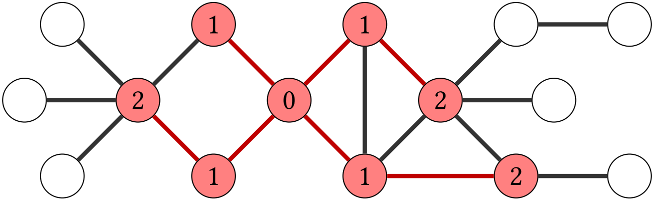


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

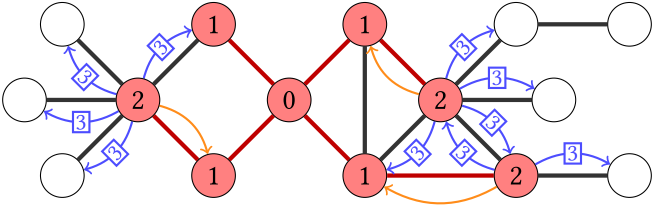


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

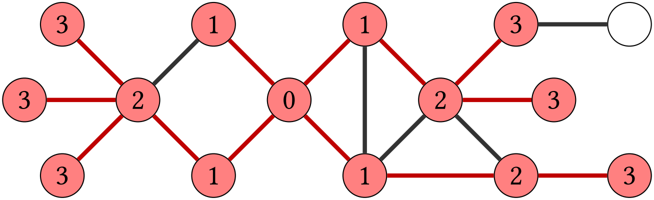


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

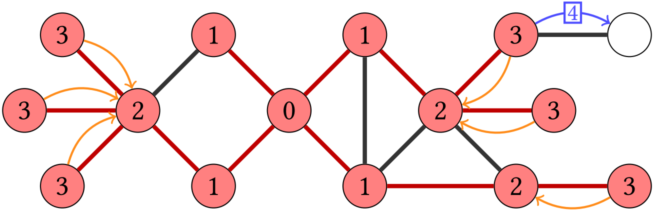


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

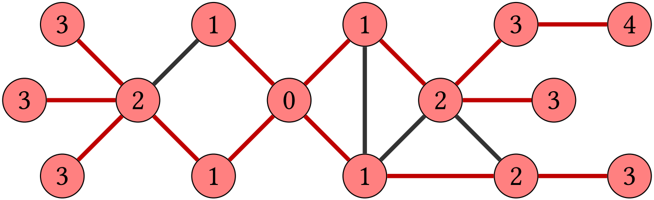


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

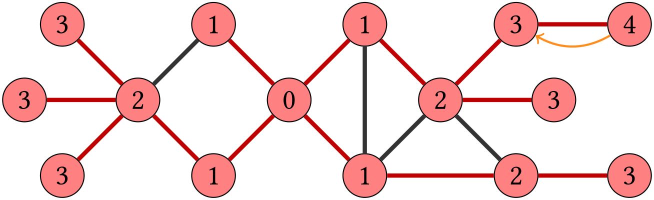


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind“-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn

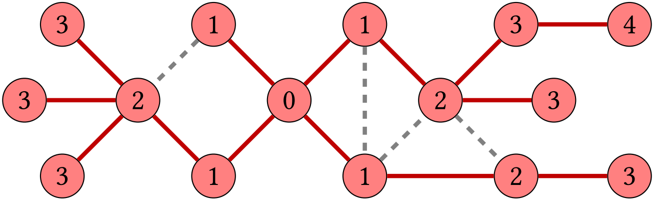


Distanz-Berechnung

Idee: Breitensuche induziert Distanzen zu s

Algorithmus:

- s hat Distanz 0 zu sich selbst und sendet in Runde 1 Distanz-Information 1 an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Distanz-Information d empfängt:
  - v speichert einen der Sender von d als Elternknoten
  - v sendet „Kind"-Nachricht an Elternknoten und Distanz-Information $d + 1$ an alle anderen Nachbarn



Zusammenfassung Breitensuche

Breitensuche mit Startknoten s:

- #Runden: $O(\mathrm{Ecc}(s)) = O(D)$
- #Nachrichten: $O(m)$

Zusammenfassung Breitensuche

Breitensuche mit Startknoten s:

- #Runden: $O(\mathrm{Ecc}(s)) = O(D)$
- #Nachrichten: $O(m)$
- Jeder Knoten v kennt Distanz $\mathrm{dist}(s, v)$

Zusammenfassung Breitensuche

Breitensuche mit Startknoten s:

- #Runden: $O(\mathrm{Ecc}(s)) = O(D)$
- #Nachrichten: $O(m)$
- Jeder Knoten v kennt Distanz $\mathrm{dist}(s, v)$
- Breitensuchbaum:
  - Jeder Knoten $u \neq s$ hat einen Elternknoten v mit $\mathrm{dist}(s, v) = \mathrm{dist}(s, u) - 1$

Zusammenfassung Breitensuche

Breitensuche mit Startknoten s:

- #Runden: $O(\mathrm{Ecc}(s)) = O(D)$
- #Nachrichten: $O(m)$
- Jeder Knoten v kennt Distanz $\mathrm{dist}(s, v)$
- Breitensuchbaum:
  - Jeder Knoten $u \neq s$ hat einen Elternknoten v mit $\mathrm{dist}(s, v) = \mathrm{dist}(s, u) - 1$
  - Jeder Knoten kennt Elternknoten und Kinder im Baum

Zusammenfassung Breitensuche

Breitensuche mit Startknoten s:

- #Runden: $O(\mathrm{Ecc}(s)) = O(D)$
- #Nachrichten: $O(m)$
- Jeder Knoten v kennt Distanz $\mathrm{dist}(s, v)$
- Breitensuchbaum:
  - Jeder Knoten $u \neq s$ hat einen Elternknoten v mit $\mathrm{dist}(s, v) = \mathrm{dist}(s, u) - 1$
  - Jeder Knoten kennt Elternknoten und Kinder im Baum
  - Jeder Breitensuchbaum ist auch ein Spannbaum

Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet

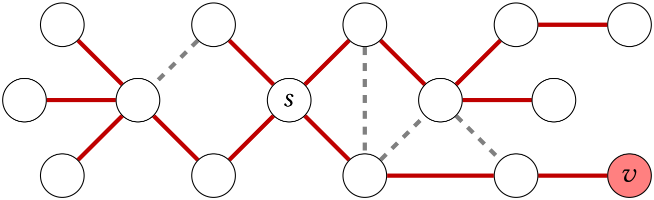
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



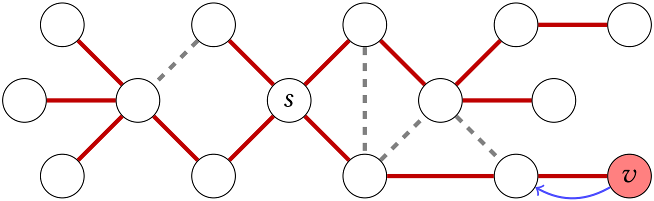
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



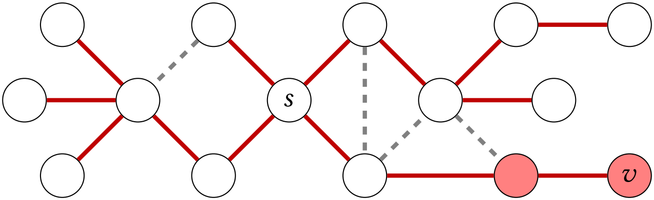
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



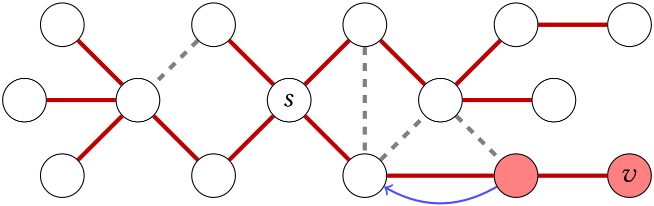
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



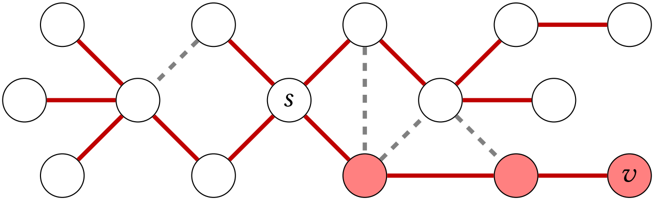
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



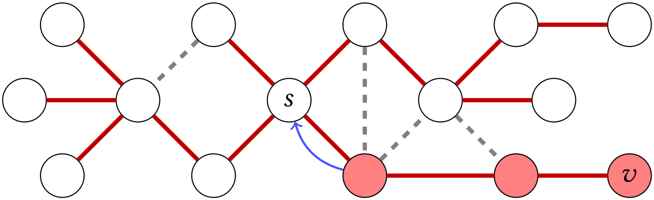
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



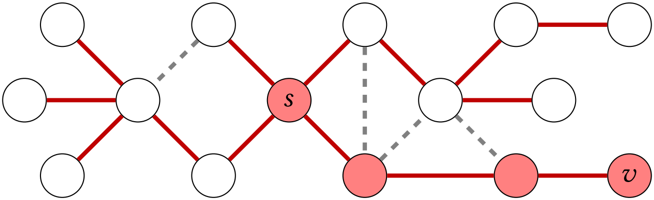
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



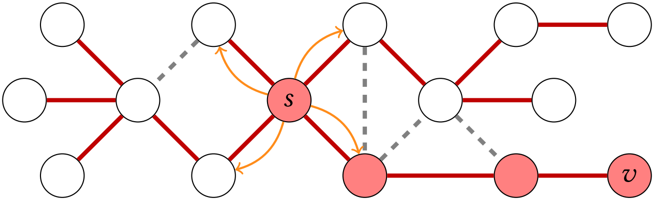
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



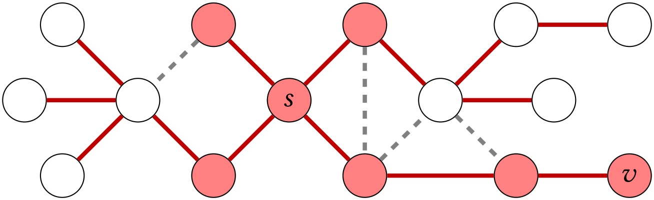
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



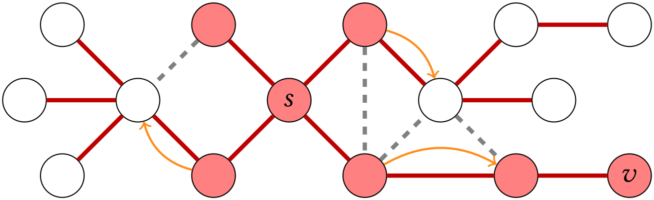
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



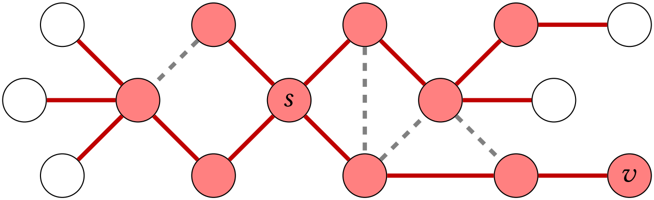
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



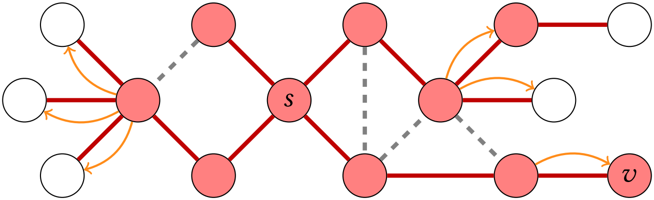
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



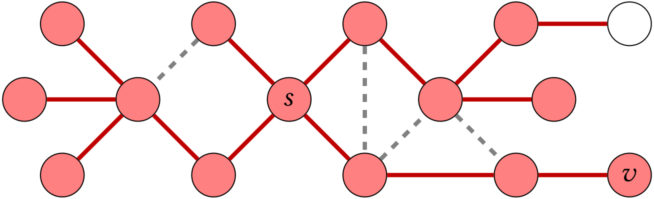
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



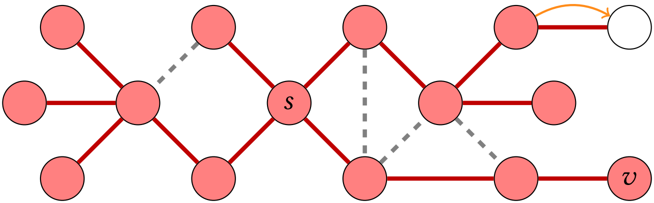
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



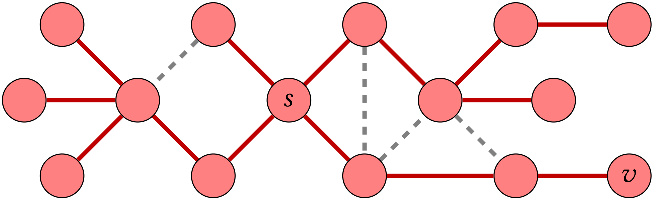
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



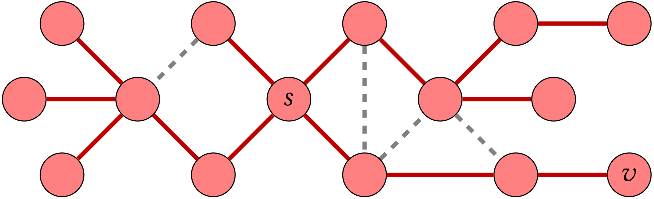
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



#Runden: $O(D)$

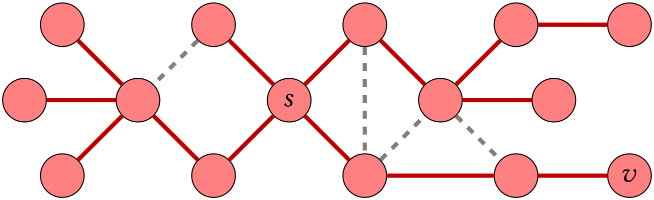
Upcast+Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten v hat Information I (der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information I

Algorithmus:

- Upcast: I wird über Elternknoten zu s gesendet
- Downcast: I wird von s aus über Kinder zu allen Knoten gesendet



#Runden: $O(D)$
#Nachrichten: $O(n)$ (statt $O(m)$ wie bei Broadcast)

Pipelining

Multipler Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten s hat k Information $I_1, \dots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information $I_1, \dots, I_k$

Multipler Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten s hat k Information $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information $I_1, \ldots, I_k$

Achtung: Limitierte Nachrichtengröße von $O(\log n)$ verhindert Bündeln zu einer einzigen Nachricht!

Multipler Downcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten s hat k Information $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Alle Knoten haben Information $I_1, \ldots, I_k$

Achtung: Limitierte Nachrichtengröße von $O(\log n)$ verhindert Bündeln zu einer einzigen Nachricht!

Naive Lösung:

- Verwende Downcast-Algorithmus k-mal
- Laufzeit: $O(kD)$

Pipeline-Algorithmus

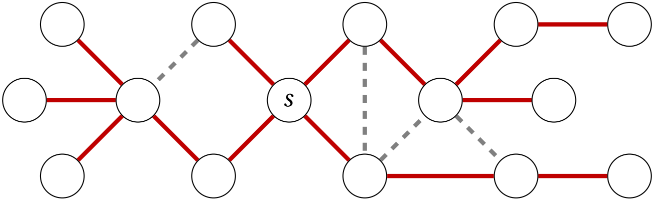
Algorithmus:

- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn

Pipeline-Algorithmus

Algorithmus:

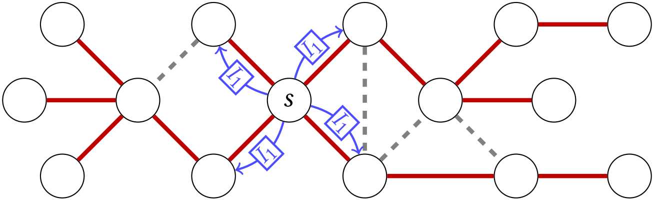
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

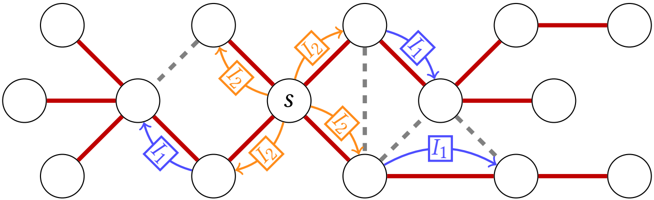
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

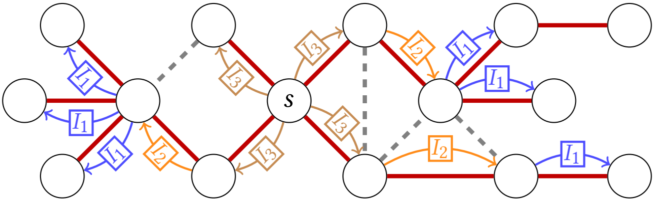
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

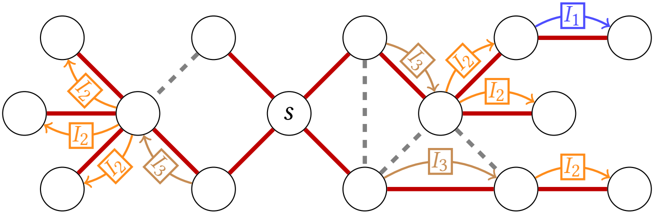
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

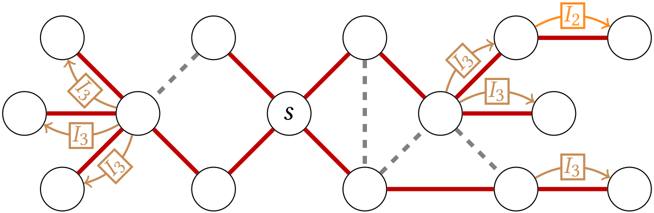
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

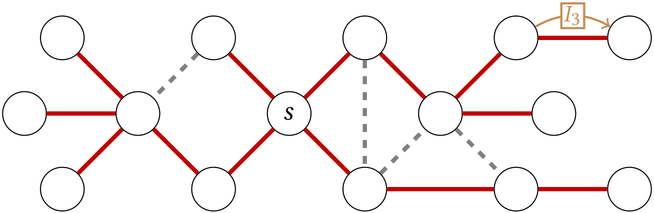
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

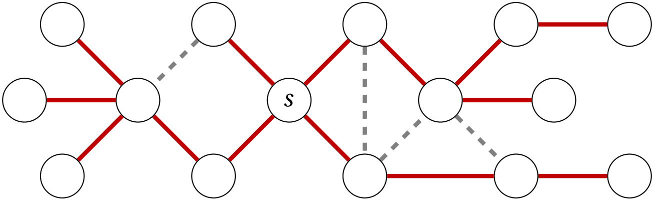
- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



Pipeline-Algorithmus

Algorithmus:

- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn



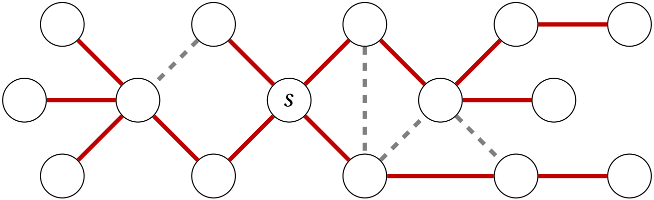
Induktionshypothese

Für jedes $1 \leq j \leq k$ und jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + j$ die Information I_j.

Pipeline-Algorithmus

Algorithmus:

- s sendet in Runde $j \leq k$ Information I_j an alle Nachbarn
- Immer wenn ein Knoten $v \neq s$ eine Information I das erste Mal empfängt: v sendet I an alle Nachbarn

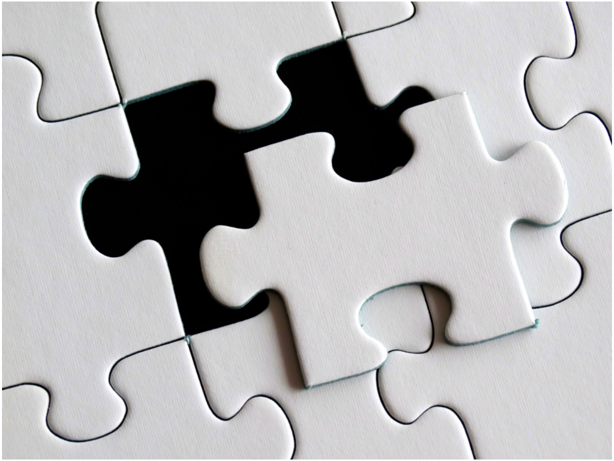


Induktionshypothese

Für jedes $1 \leq j \leq k$ und jeden Knoten $v \neq s$ gilt: Knoten v empfängt spätestens in Runde $\mathrm{dist}(s, v) + j$ die Information I_j.

$\Rightarrow$ **Laufzeit** $O(\mathrm{Ecc}(s) + k) = O(D + k)$

Aggregation



Convergecast

Ziel: s möchte sicherstellen, dass Broadcast/Breitensuche abgeschlossen ist

Convergecast

Ziel: s möchte sicherstellen, dass Broadcast/Breitensuche abgeschlossen ist

Idee: Sende ACK-Nachricht von jedem Knoten an s per Upcast

Convergecast

Ziel: s möchte sicherstellen, dass Broadcast/Breitensuche abgeschlossen ist

Idee: Sende ACK-Nachricht von jedem Knoten an s per Upcast

Naiver Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Jeder Knoten sendet ACK-Nachricht an s per Upcast im Baum

Convergecast

Ziel: s möchte sicherstellen, dass Broadcast/Breitensuche abgeschlossen ist

Idee: Sende ACK-Nachricht von jedem Knoten an s per Upcast

Naiver Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Jeder Knoten sendet ACK-Nachricht an s per Upcast im Baum

Problem: Congestion trotz Pipelining

Convergecast

Ziel: s möchte sicherstellen, dass Broadcast/Breitensuche abgeschlossen ist

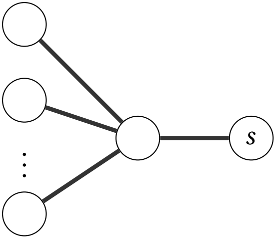
Idee: Sende ACK-Nachricht von jedem Knoten an s per Upcast

Naiver Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Jeder Knoten sendet ACK-Nachricht an s per Upcast im Baum

Problem: Congestion trotz Pipelining

$\Theta(n)$ Runden im Worst Case!



Aggregation der ACK-Nachrichten

Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus

Aggregation der ACK-Nachrichten

Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten

Aggregation der ACK-Nachrichten

Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen

Aggregation der ACK-Nachrichten

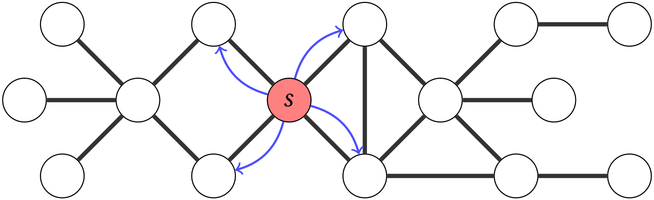
Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen

Aggregation der ACK-Nachrichten

Algorithmus:

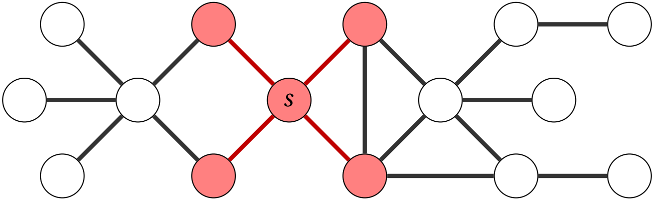
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

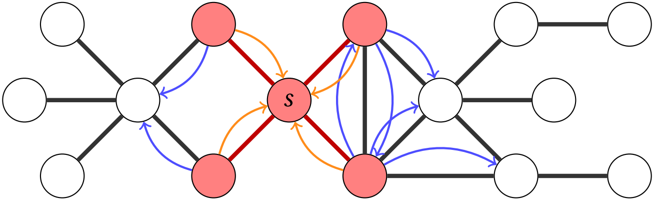
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

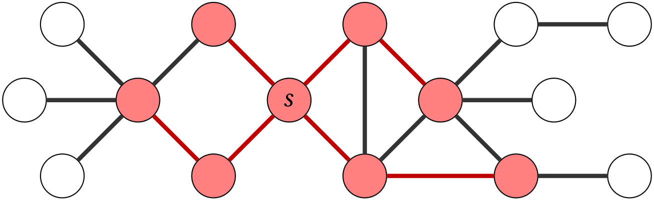
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

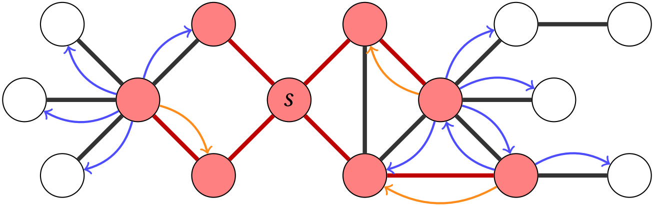
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

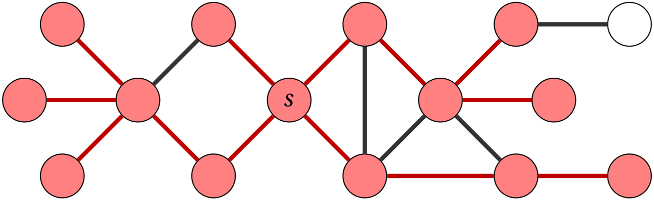
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

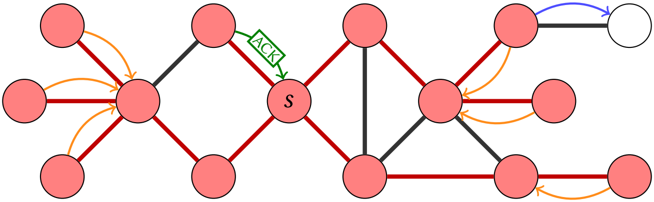
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

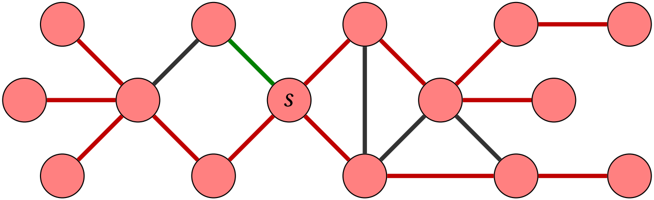
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

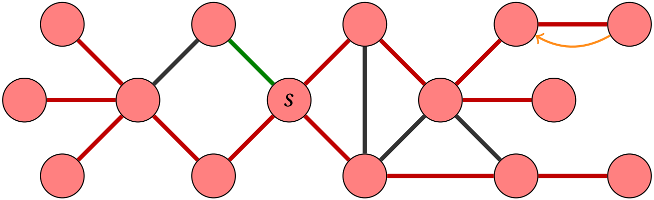
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

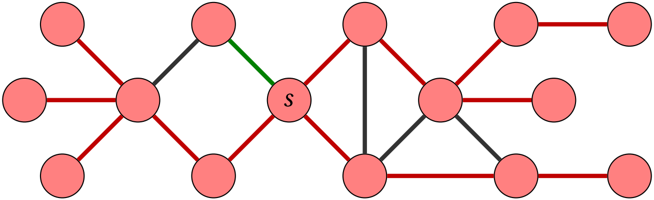
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

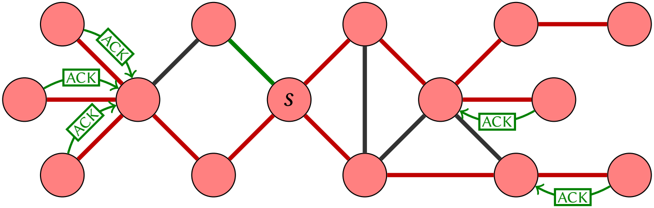
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

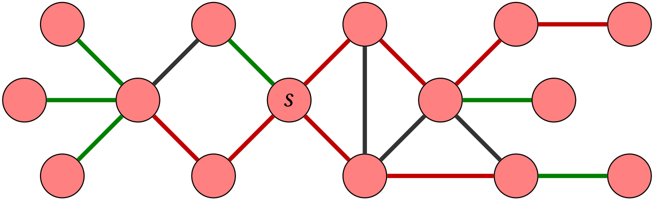
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

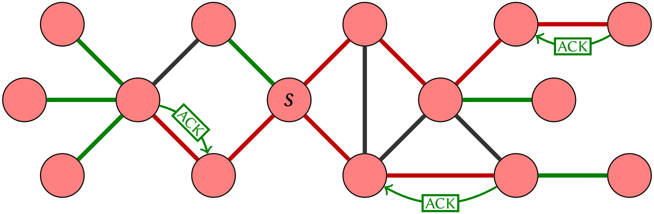
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

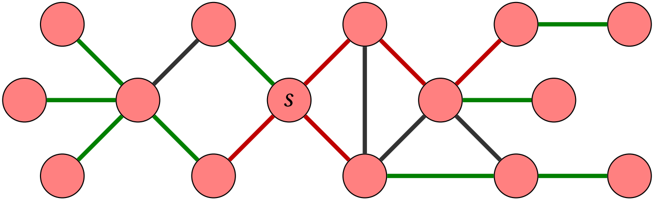
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

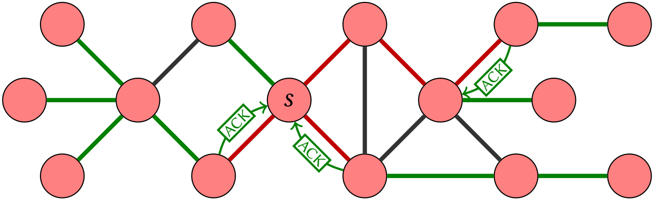
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

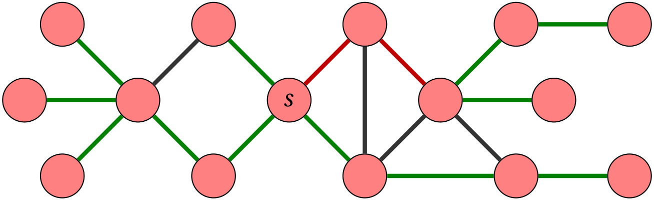
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

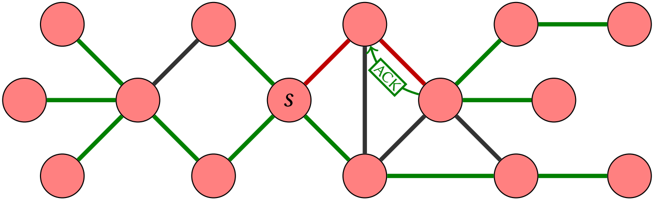
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

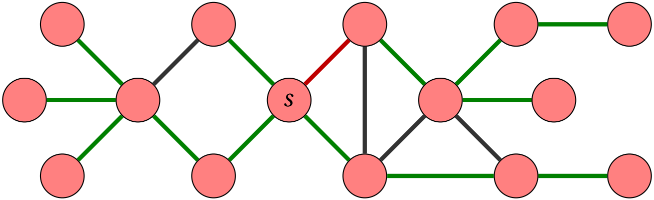
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

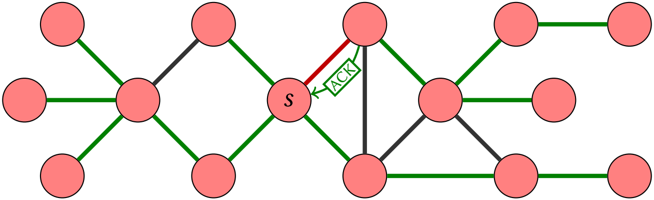
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

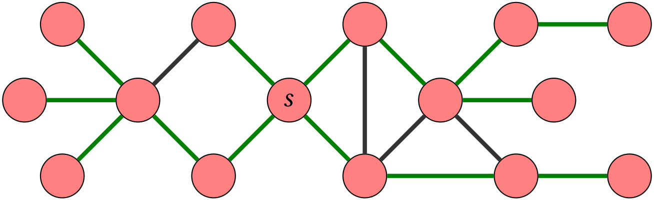
- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregation der ACK-Nachrichten

Algorithmus:

- Führe Breitensuche mit Spannbaumberechnung aus
- Sobald Knoten ACK von allen Kindern empfangen hat: Sende ACK an Elternknoten
- Blätter im Baum erkennen sich selbst als solche, wenn sie keine Kind-Nachricht empfangen
- s wartet bis ACK von allen Kindern empfangen



Aggregatfunktionen

- Funktionale Sicht: ACK-Bit eines Knotens ist Und-Verknüpfung der ACK-Bits der Kinder

Aggregatfunktionen

- Funktionale Sicht: ACK-Bit eines Knotens ist Und-Verknüpfung der ACK-Bits der Kinder
- Baumstruktur ermöglicht Bottom-Up Berechnung und Warten auf Funktionswerte der Kinder

Aggregatfunktionen

- Funktionale Sicht: ACK-Bit eines Knotens ist Und-Verknüpfung der ACK-Bits der Kinder
- Baumstruktur ermöglicht Bottom-Up Berechnung und Warten auf Funktionswerte der Kinder
- Weitere Aggregatfunktionen: Summe, Minimum, Maximum, etc.

Queuing

Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \dots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \dots, I_k$

Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten

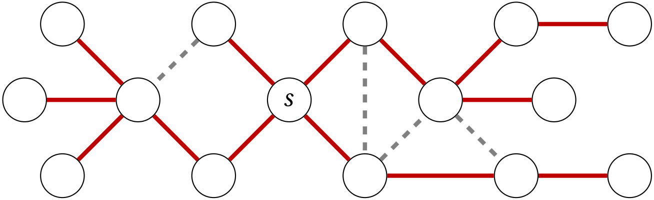
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



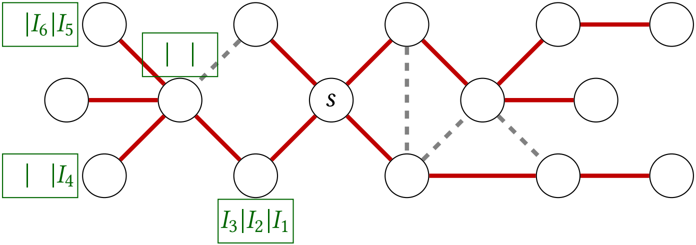
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



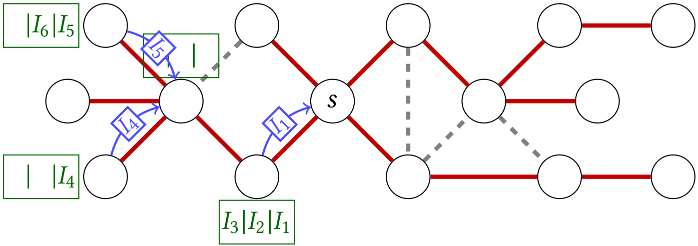
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



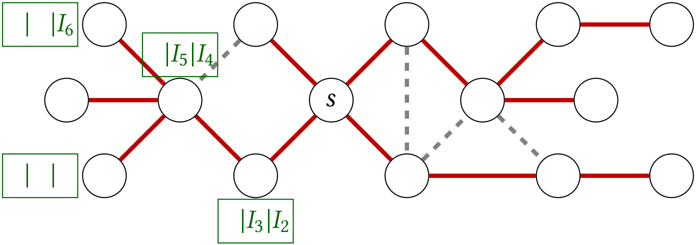
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



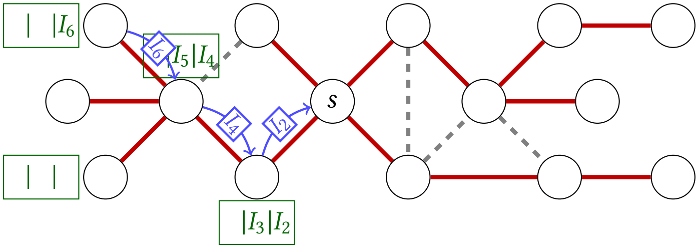
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



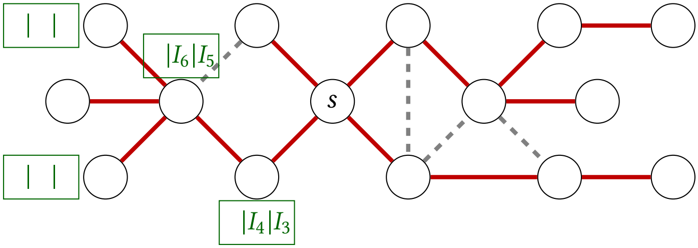
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



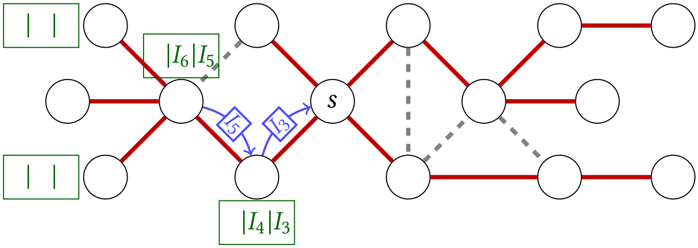
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



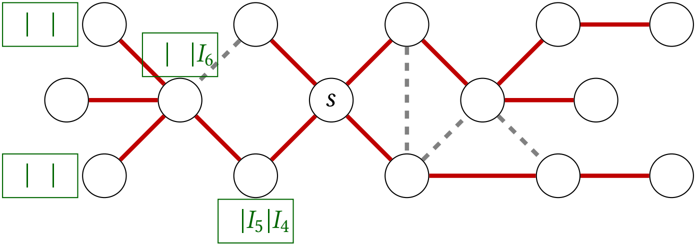
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



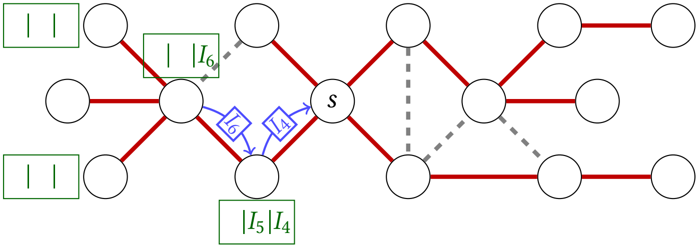
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



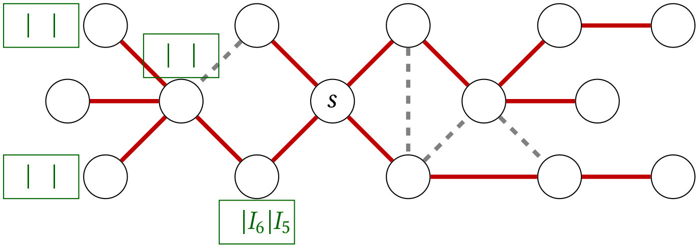
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



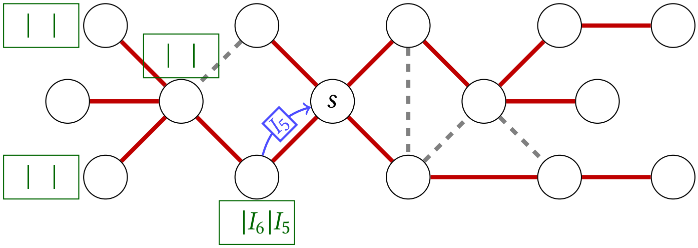
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



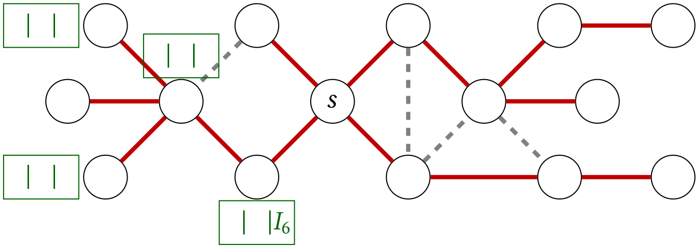
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



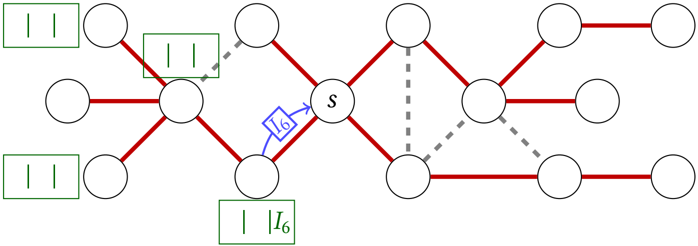
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



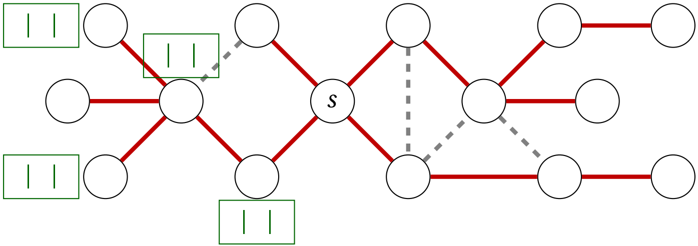
Multipler Upcast

Annahme: Breitensuchbaum von s wurde bereits berechnet

Ausgangssituation: Knoten das Netzwerks haben k Informationen $I_1, \ldots, I_k$ (jeweils der Größe $O(\log n)$)
Ziel: Knoten s hat Informationen $I_1, \ldots, I_k$

Algorithmus:

- Jeder Knoten hat eine Queue noch nicht weitergeleiteter Informationen
- In jeder Runde, sendet jeder Knoten eine der Informationen aus seiner Queue an seinen Elternknoten



Laufzeitschranke mit Hilfslemma

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Laufzeitschranke mit Hilfslemma

Lemma

Für jedes $r \geq 1$ *und jeden Knoten* v *mit* $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ *gilt: Wenn* v *keine Information in Runde* r *empfängt, dann auch nicht in Runde* $r + 1$.

Theorem

Der Algorithmus für den Multiplen Upcast benötigt $O(D + k)$ *Runden*

Laufzeitschranke mit Hilfslemma

Lemma

Für jedes $r \geq 1$ *und jeden Knoten* v *mit* $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ *gilt: Wenn* v *keine Information in Runde* r *empfängt, dann auch nicht in Runde* $r + 1$.

Theorem

Der Algorithmus für den Multiplen Upcast benötigt $O(D + k)$ *Runden*

Beweis:

- Ab Runde $r = \mathrm{Ecc}(s) + 1$ gilt wegen $\mathrm{dist}(s, v) = 0 \geq \mathrm{Ecc}(s) - r + 1$:
  s muss in jeder Runde eine Information empfangen, oder s empfängt überhaupt keine Informationen mehr und der Algorithmus terminiert

Laufzeitschranke mit Hilfslemma

Lemma

Für jedes $r \geq 1$ *und jeden Knoten* v *mit* $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ *gilt: Wenn* v *keine Information in Runde* r *empfängt, dann auch nicht in Runde* $r + 1$.

Theorem

Der Algorithmus für den Multiplen Upcast benötigt $O(D + k)$ *Runden*

Beweis:

- Ab Runde $r = \mathrm{Ecc}(s) + 1$ gilt wegen $\mathrm{dist}(s, v) = 0 \geq \mathrm{Ecc}(s) - r + 1$:
  s muss in jeder Runde eine Information empfangen, oder s empfängt überhaupt keine Informationen mehr und der Algorithmus terminiert
- Jede der k Informationen wird von jedem Knoten (und damit auch von s) höchstens einmal empfangen

Laufzeitschranke mit Hilfslemma

Lemma

Für jedes $r \geq 1$ *und jeden Knoten* v *mit* $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ *gilt: Wenn* v *keine Information in Runde* r *empfängt, dann auch nicht in Runde* $r + 1$*.*

Theorem

Der Algorithmus für den Multiplen Upcast benötigt $O(D + k)$ *Runden*

Beweis:

- Ab Runde $r = \mathrm{Ecc}(s) + 1$ gilt wegen $\mathrm{dist}(s, v) = 0 \geq \mathrm{Ecc}(s) - r + 1$: s muss in jeder Runde eine Information empfangen, oder s empfängt überhaupt keine Informationen mehr und der Algorithmus terminiert
- Jede der k Informationen wird von jedem Knoten (und damit auch von s) höchstens einmal empfangen
- **Somit:** Höchstens $\mathrm{Ecc}(s) + k = O(D + k)$ Runden

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\mathrm{dist}(s,s) \geq \mathrm{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r+1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\mathrm{dist}(s,v) \geq \mathrm{Ecc}(s) - r + 1 = \mathrm{Ecc}(s)$

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\mathrm{dist}(s, s) \geq \mathrm{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1 = \mathrm{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - ★ Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - ★ Es gilt: $\operatorname{dist}(s, u) = \operatorname{dist}(s, v) + 1$

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\text{dist}(s, s) \geq \text{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\text{dist}(s, v) \geq \text{Ecc}(s) - r + 1 = \text{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\text{dist}(s, v) \geq \text{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - ★ Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - ★ Es gilt: $\text{dist}(s, u) = \text{dist}(s, v) + 1 \geq \text{Ecc}(s) - r + 1 + 1$

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ *und jeden Knoten* v *mit* $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ *gilt: Wenn* v *keine Information in Runde* r *empfängt, dann auch nicht in Runde* $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - ★ Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - ★ Es gilt: $\operatorname{dist}(s, u) = \operatorname{dist}(s, v) + 1 \geq \operatorname{Ecc}(s) - r + 1 + 1 = \operatorname{Ecc}(s) - (r - 1) + 1$

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\mathrm{dist}(s, s) \geq \mathrm{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1 = \mathrm{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - Es gilt: $\mathrm{dist}(s, u) = \mathrm{dist}(s, v) + 1 \geq \mathrm{Ecc}(s) - r + 1 + 1 = \mathrm{Ecc}(s) - (r - 1) + 1$
    - Nach IH: u empfängt keine Information in Runde $(r - 1) + 1 = r$

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\operatorname{dist}(s, s) \geq \operatorname{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1 = \operatorname{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\operatorname{dist}(s, v) \geq \operatorname{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - ★ Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - ★ Es gilt: $\operatorname{dist}(s, u) = \operatorname{dist}(s, v) + 1 \geq \operatorname{Ecc}(s) - r + 1 + 1 = \operatorname{Ecc}(s) - (r - 1) + 1$
    - ★ Nach IH: u empfängt keine Information in Runde $(r - 1) + 1 = r$
    - ★ Queue von u is somit in Runde r immer noch leer und u sendet keine Information an v

Beweis des Lemmas

Lemma

Für jedes $r \geq 1$ und jeden Knoten v mit $\mathrm{dist}(s, s) \geq \mathrm{Ecc}(s) - r + 1$ gilt: Wenn v keine Information in Runde r empfängt, dann auch nicht in Runde $r + 1$.

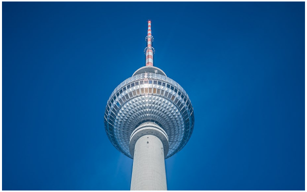
Beweis:

- Induktionsbasis ($r = 1$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1 = \mathrm{Ecc}(s)$
  - Dann ist v ein Blatt im Breitensuchbaum und empfängt keine Information
- Induktionsschritt ($r \geq 2$):
  - Sei v ein Knoten mit $\mathrm{dist}(s, v) \geq \mathrm{Ecc}(s) - r + 1$ (der kein Blatt ist)
  - Sei u ein Kind von v im Breitensuchbaum
    - ★ Da v in Runde r keine Information empfängt, hat u in Runde $r - 1$ keine Information an v gesendet
    - ★ Deshalb hat u in Runde $r - 1$ leere Queue und keine Information empfangen
    - ★ Es gilt: $\mathrm{dist}(s, u) = \mathrm{dist}(s, v) + 1 \geq \mathrm{Ecc}(s) - r + 1 + 1 = \mathrm{Ecc}(s) - (r - 1) + 1$
    - ★ Nach IH: u empfängt keine Information in Runde $(r - 1) + 1 = r$
    - ★ Queue von u is somit in Runde r immer noch leer und u sendet keine Information an v
  - **Somit:** v empfängt keine Information in Runde $r + 1$

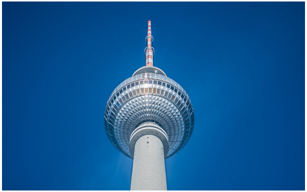
Zusammenfassung

Zusammenfassung

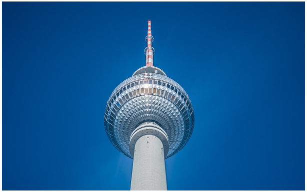
Zusammenfassung



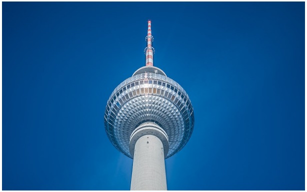
Zusammenfassung



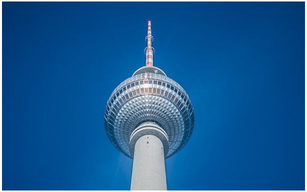
Zusammenfassung



Zusammenfassung



Zusammenfassung



Quellen

Der Inhalt dieser Vorlesungseinheit basiert zum Teil auf einer Vorlesungseinheit von Danupon Nanongkai.

Literatur:

- David Peleg (2000) *Distributed Computing*, Kapitel 3 u. 4, SIAM.