Gerichtete Graphen

Nico Pistel

Diskrete Mathematik und Stochastik, 2019

Fachbereich Wirtschaft und Informationstechnik
Westfälische Hochschule Bocholt

Outline

Begriffe & Definitionen

Begriffe & Definitionen

- Ein gerichteter Graph (**Di**rected **Graph** - *Digraph*) ist ein Graph, wobei die Kanten mit einer Richtung ausgezeichnet sind

Bereits zur Visualisierung von binären Relationen bekannt. Gerichtete Kannten werden auch Arrows, Pfeile oder auch nur Kanten genannt (aus Zusammenhang schließen)

Begriffe & Definitionen

- Ein gerichteter Graph (**Di**rected **Graph** - *Digraph*) ist ein Graph, wobei die Kanten mit einer Richtung ausgezeichnet sind
- Unterschied zum ungerichteten Graphen: Kantenrelation E von $G = (V, E)$ ist nicht zwingend symmetrisch

Bereits zur Visualisierung von binären Relationen bekannt. Gerichtete Kannten werden auch Arrows, Pfeile oder auch nur Kanten genannt (aus Zusammenhang schließen)

Begriffe & Definitionen

- Ein gerichteter Graph (**Di**rected **Graph** - *Digraph*) ist ein Graph, wobei die Kanten mit einer Richtung ausgezeichnet sind
- Unterschied zum ungerichteten Graphen: Kantenrelation E von $G = (V, E)$ ist nicht zwingend symmetrisch
  - Adjazenzmatrix $\mathbf{M}$ von G ist nicht zwingend symmetrisch

Bereits zur Visualisierung von binären Relationen bekannt. Gerichtete Kannten werden auch Arrows, Pfeile oder auch nur Kanten genannt (aus Zusammenhang schließen)

Begriffe & Definitionen

- Ein gerichteter Graph (**Di**rected **Graph** - *Digraph*) ist ein Graph, wobei die Kanten mit einer Richtung ausgezeichnet sind
- Unterschied zum ungerichteten Graphen: Kantenrelation E von $G = (V, E)$ ist nicht zwingend symmetrisch
  - Adjazenzmatrix $\mathbf{M}$ von G ist nicht zwingend symmetrisch
- Die Richtung der gerichtete Kanten werden durch Pfeile gekennzeichnet

Bereits zur Visualisierung von binären Relationen bekannt. Gerichtete Kannten werden auch Arrows, Pfeile oder auch nur Kanten genannt (aus Zusammenhang schließen)

Begriffe & Definitionen

- Ein schlichter (Di-)Graph hat keine Schlingen und keine mehrfachen gerichteten Kanten

Schlichte (und natürlich ENDLICHE) Digraphen werden betrachtet

Begriffe & Definitionen

- Ein schlichter (Di-)Graph hat keine Schlingen und keine mehrfachen gerichteten Kanten
- Anzahl der ausgehenden Kanten von einem Knoten v ist der Ausgangsgrad (*Outdegree*) von v: $\deg^+(v)$

Schlichte (und natürlich ENDLICHE) Digraphen werden betrachtet

Begriffe & Definitionen

- Ein schlichter (Di-)Graph hat keine Schlingen und keine mehrfachen gerichteten Kanten
- Anzahl der ausgehenden Kanten von einem Knoten v ist der Ausgangsgrad (*Outdegree*) von v: $\deg^+(v)$
- Anzahl der eingehenden Kanten an einem Knoten v ist der Eingangsgrad (*Indegree*) von v: $\deg^-(v)$

Schlichte (und natürlich ENDLICHE) Digraphen werden betrachtet

Begriffe & Definitionen

- Ein schlichter (Di-)Graph hat keine Schlingen und keine mehrfachen gerichteten Kanten
- Anzahl der ausgehenden Kanten von einem Knoten v ist der Ausgangsgrad (*Outdegree*) von v: $\deg^+(v)$
- Anzahl der eingehenden Kanten an einem Knoten v ist der Eingangsgrad (*Indegree*) von v: $\deg^-(v)$

Lemma

$$\sum_{v \in V} \deg^+(v) = \sum_{v \in V} \deg^-(v) = |E|$$

Schlichte (und natürlich ENDLICHE) Digraphen werden betrachtet

Begriffe & Definitionen

- Ein Digraph ist (schwach) zusammenhängend, wenn der zugrunde liegende (ungerichtete) Graph zusammenhängend ist

Begriffe & Definitionen

- Ein Digraph ist (schwach) zusammenhängend, wenn der zugrunde liegende (ungerichtete) Graph zusammenhängend ist
  - Er ist stark zusammenhängend, wenn es zu jeden Knotenpaar (u, v) einen Weg von u nach v gibt und einen Weg von v nach u

Begriffe & Definitionen

- $(u, v) \in E \Longrightarrow u$ ist ein Vorgänger von v

Begriffe & Definitionen

- $(u, v) \in E \Longrightarrow u$ ist ein Vorgänger von v
- Ein Digraph ohne Zyklen heißt azyklischer Digraph

Begriffe & Definitionen

- $(u, v) \in E \Longrightarrow u$ ist ein Vorgänger von v
- Ein Digraph ohne Zyklen heißt azyklischer Digraph
  - **D**irected **A**cyclic **G**raph (*DAG*)

Begriffe & Definitionen

- $(u, v) \in E \Longrightarrow u$ ist ein Vorgänger von v
- Ein Digraph ohne Zyklen heißt azyklischer Digraph
  - **D**irected **A**cyclic **G**raph (*DAG*)
  - Bei Problemen der Aufgabenplanung heißt ein DAG auch ein PERT-Chart
    - Program Evaluation and Review Technique

Topologisches Sortieren

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden
  - Problem: Aufgaben setzen ggf. voraus, dass andere Aufgaben bereits erledigt wurden

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden
  - Problem: Aufgaben setzen ggf. voraus, dass andere Aufgaben bereits erledigt wurden
  - Motivation: Scheduling, Build-Prozesse

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden
  - Problem: Aufgaben setzen ggf. voraus, dass andere Aufgaben bereits erledigt wurden
  - Motivation: Scheduling, Build-Prozesse
- Problemstellung lässt sich als DAG modellieren

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden
  - Problem: Aufgaben setzen ggf. voraus, dass andere Aufgaben bereits erledigt wurden
  - Motivation: Scheduling, Build-Prozesse
- Problemstellung lässt sich als DAG modellieren
  - Warum azyklisch und nicht als allgemeiner (gerichteter) Graph?

Topologisches Sortieren

- Beispiel: Aufgaben sollen abgearbeitet werden
  - Problem: Aufgaben setzen ggf. voraus, dass andere Aufgaben bereits erledigt wurden
  - Motivation: Scheduling, Build-Prozesse
- Problemstellung lässt sich als DAG modellieren
  - Warum azyklisch und nicht als allgemeiner (gerichteter) Graph?
    - Zyklen deuten auf Aufgaben hin, welche sich selber zur Abarbeitung voraussetzen
    - Solche Aufgaben könnten nie abgearbeitet werden

Topologisches Sortieren

- Gesucht ist eine Anordnung, die eine Reihenfolge vorgibt, in welcher die Aufgaben abgearbeitet werden können
  - So, dass jede Aufgabe direkt bearbeitet werden kann (alle nötigen Vorausetzungen wurden zuvor erfüllt)

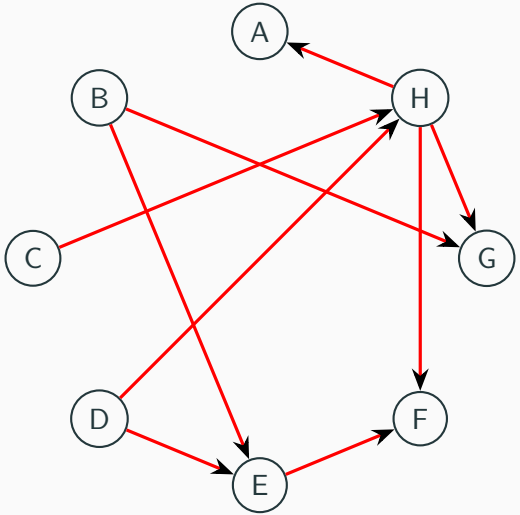
Topologisches Sortieren

- Gesucht ist eine Anordnung, die eine Reihenfolge vorgibt, in welcher die Aufgaben abgearbeitet werden können
  - So, dass jede Aufgabe direkt bearbeitet werden kann (alle nötigen Vorausetzungen wurden zuvor erfüllt)
- Eine topologische Sortierung liefert eine solche *konsistente Anordnung* der Knoten

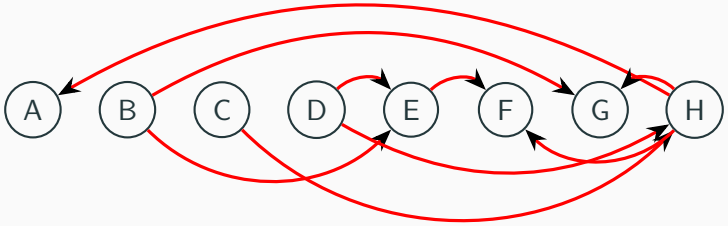
Topologisches Sortieren

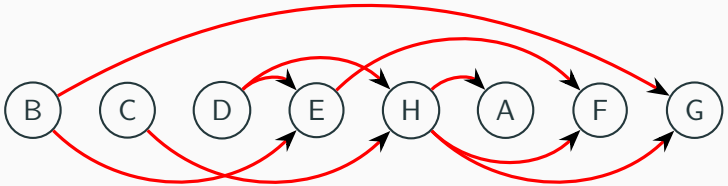
- Gesucht ist eine Anordnung, die eine Reihenfolge vorgibt, in welcher die Aufgaben abgearbeitet werden können
  - So, dass jede Aufgabe direkt bearbeitet werden kann (alle nötigen Vorausetzungen wurden zuvor erfüllt)
- Eine topologische Sortierung liefert eine solche *konsistente Anordnung* der Knoten
- Für alle Kanten $(u, v) \in E$ gilt, dass u in der Anordnung stets vor v kommt

Beispiel: Topologisches Sortieren



Beispiel: Topologisches Sortieren





Kahn's Algorithmus

- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus

Kahn's Algorithmus

- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus
- Wechsel der Datenstruktur: Für jeden Knoten v wird die Menge seiner Vorgänger $A(v)$ betrachtet

Kahn's Algorithmus

- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus
- Wechsel der Datenstruktur: Für jeden Knoten v wird die Menge seiner Vorgänger $A(v)$ betrachtet
- Anordnung wird intuitiv erzeugt: In jedem Schritt wird ein Knoten ausgewählt, welcher direkt abgearbeitet werden kann (mögliche Vorgänger sind bereits in der Anordnung vorhanden)

Kahn's Algorithmus

- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus
- Wechsel der Datenstruktur: Für jeden Knoten v wird die Menge seiner Vorgänger $A(v)$ betrachtet
- Anordnung wird intuitiv erzeugt: In jedem Schritt wird ein Knoten ausgewählt, welcher direkt abgearbeitet werden kann (mögliche Vorgänger sind bereits in der Anordnung vorhanden)
- Wiederholung bis alle Knoten in der Anordnung vorhanden sind

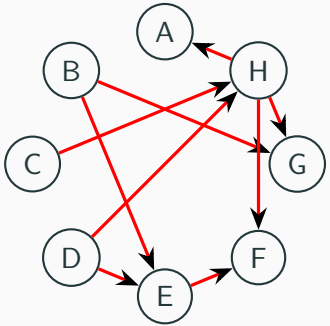
Kahn's Algorithmus

- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus
- Wechsel der Datenstruktur: Für jeden Knoten v wird die Menge seiner Vorgänger $A(v)$ betrachtet
- Anordnung wird intuitiv erzeugt: In jedem Schritt wird ein Knoten ausgewählt, welcher direkt abgearbeitet werden kann (mögliche Vorgänger sind bereits in der Anordnung vorhanden)
- Wiederholung bis alle Knoten in der Anordnung vorhanden sind
- Es lässt sich immer (mindestens) ein solch ein Knoten auswählen
  - Sonst ist ein Zyklus im Graph vorhanden

Kahn's Algorithmus

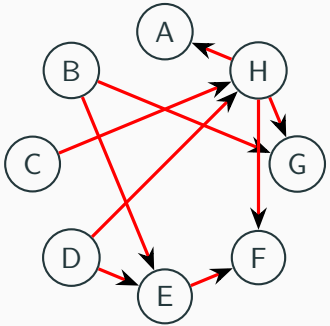
- Ein möglicher Algorithmus zur topologischen Sortierung ist Kahn's Algorithmus
- Wechsel der Datenstruktur: Für jeden Knoten v wird die Menge seiner Vorgänger $A(v)$ betrachtet
- Anordnung wird intuitiv erzeugt: In jedem Schritt wird ein Knoten ausgewählt, welcher direkt abgearbeitet werden kann (mögliche Vorgänger sind bereits in der Anordnung vorhanden)
- Wiederholung bis alle Knoten in der Anordnung vorhanden sind
- Es lässt sich immer (mindestens) ein solch ein Knoten auswählen
  - Sonst ist ein Zyklus im Graph vorhanden
- Anordnung muss nicht eindeutig sein
  - Es können in einem Schritt mehrere Knoten zur Auswahl stehen

Beispiel: Kahn's Algorithmus



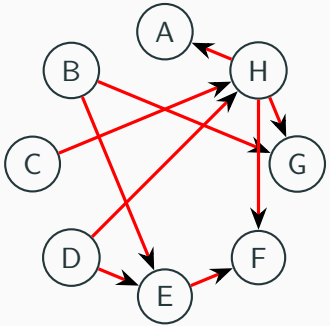
i	v	$A(v)$
	A	
	B	
	C	
	D	
	E	
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



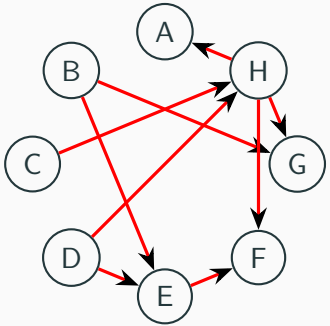
i	v	$A(v)$
	A	{H}
	B	
	C	
	D	
	E	
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



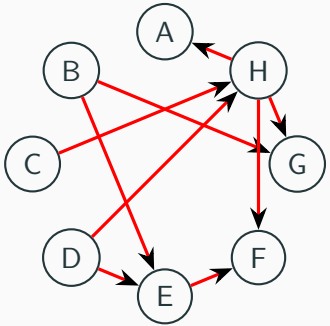
i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	
	D	
	E	
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



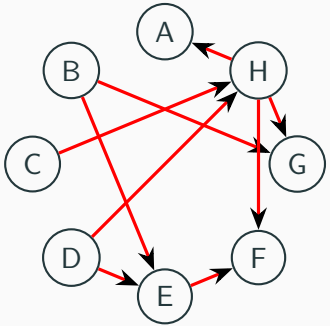
i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	
	E	
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



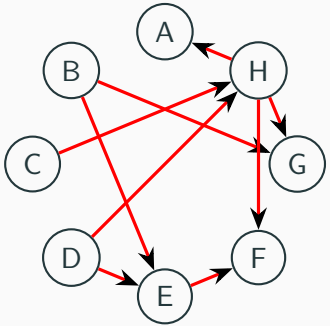
i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



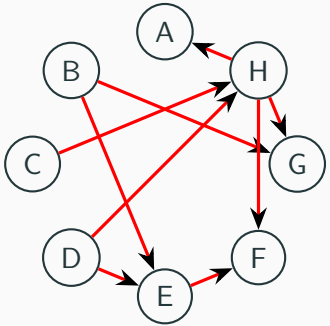
i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	{B, D}
	F	
	G	
	H	

Beispiel: Kahn's Algorithmus



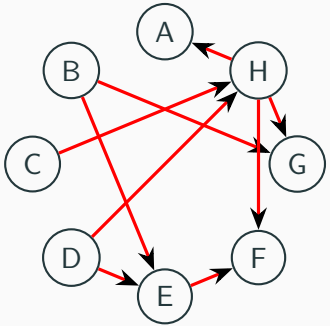
i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	{B, D}
	F	{E, H}
	G	
	H	

Beispiel: Kahn's Algorithmus



i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	{B, D}
	F	{E, H}
	G	{B, H}
	H	

Beispiel: Kahn's Algorithmus



i	v	$A(v)$
	A	{H}
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	{B, D}
	F	{E, H}
	G	{B, H}
	H	{C, D}

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	$\{\mathsf{B}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\mathsf{B}, \mathsf{H}\}$
	H	$\{\mathsf{C}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

B

i	v	$A(v)$
	A	{H}
1	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	{B, D}
	F	{E, H}
	G	{B, H}
	H	{C, D}

Beispiel: Kahn's Algorithmus

B

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\mathsf{C}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

B

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
	C	$\varnothing$
	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\mathsf{C}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{H\}$
1	B	$\varnothing$
2	C	$\varnothing$
	D	$\varnothing$
	E	$\{\cancel{B}, D\}$
	F	$\{E, H\}$
	G	$\{\cancel{B}, H\}$
	H	$\{C, D\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \mathsf{D}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \mathsf{D}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\mathsf{E}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

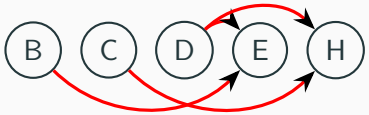
Beispiel: Kahn's Algorithmus

i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\cancel{\mathsf{E}}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

Beispiel: Kahn's Algorithmus

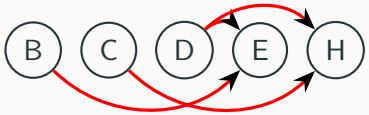
i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\cancel{\mathsf{E}}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

Beispiel: Kahn's Algorithmus



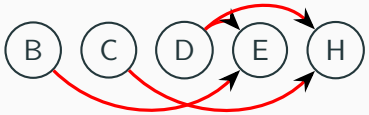
i	v	$A(v)$
	A	$\{\mathsf{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{\mathsf{B}}, \cancel{\mathsf{D}}\}$
	F	$\{\cancel{\mathsf{E}}, \mathsf{H}\}$
	G	$\{\cancel{\mathsf{B}}, \mathsf{H}\}$
5	H	$\{\cancel{\mathsf{C}}, \cancel{\mathsf{D}}\}$

Beispiel: Kahn's Algorithmus



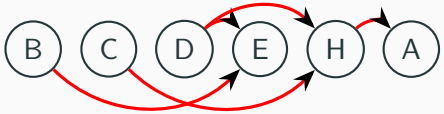
i	v	$A(v)$
	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



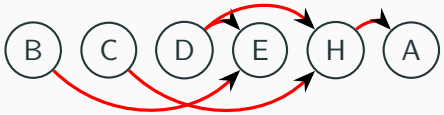
i	v	$A(v)$
	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



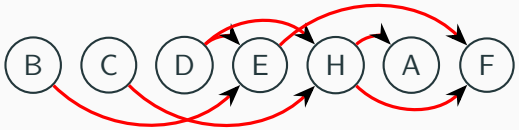
i	v	$A(v)$
6	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



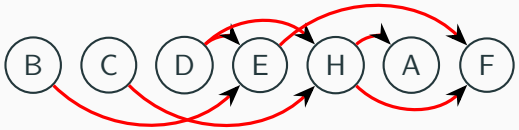
i	v	$A(v)$
6	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



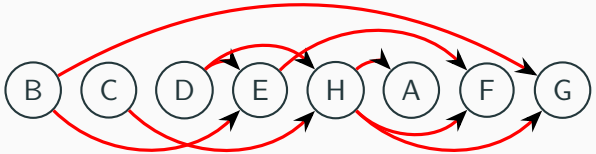
i	v	$A(v)$
6	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
7	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



i	v	$A(v)$
6	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
7	F	$\{\cancel{E}, \cancel{H}\}$
	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Beispiel: Kahn's Algorithmus



i	v	$A(v)$
6	A	$\{\cancel{H}\}$
1	B	$\varnothing$
2	C	$\varnothing$
3	D	$\varnothing$
4	E	$\{\cancel{B}, \cancel{D}\}$
7	F	$\{\cancel{E}, \cancel{H}\}$
8	G	$\{\cancel{B}, \cancel{H}\}$
5	H	$\{\cancel{C}, \cancel{D}\}$

Wege in Digraphen

Wege in Digraphen

- Existenz von Wegen zwischen beliebigen Knoten

Wege in Digraphen

- Existenz von Wegen zwischen beliebigen Knoten
- Adjazenzmatrix $\mathbf{M} \longrightarrow$ Erreichbarkeitsmatrix $\mathbf{M}^*$

Wege in Digraphen

- Existenz von Wegen zwischen beliebigen Knoten
- Adjazenzmatrix $\mathbf{M} \longrightarrow$ Erreichbarkeitsmatrix $\mathbf{M}^*$
- Kantenrelation $E \longrightarrow$ transitiver Abschluss E^*

Wege als transitiver Abschluss der Kantenrelation

- Kantenrelation E: Menge von Wegen der Länge 1 zwischen Knoten

Nicht Path sondern Walk!

Wege als transitiver Abschluss der Kantenrelation

- Kantenrelation E: Menge von Wegen der Länge 1 zwischen Knoten
- Menge von Wegen der Länge 2 als Komposition von E mit sich selber:
  $E \circ E = \{(v_1, v_2) \in V^2 | \exists u \in V : (v_1, u) \in E \wedge (u, v_2) \in E\}$

Nicht Path sondern Walk!

Wege als transitiver Abschluss der Kantenrelation

- Kantenrelation E: Menge von Wegen der Länge 1 zwischen Knoten
- Menge von Wegen der Länge 2 als Komposition von E mit sich selber:
  $E \circ E = \{(v_1, v_2) \in V^2 | \exists u \in V : (v_1, u) \in E \wedge (u, v_2) \in E\}$
- Menge von Wegen einer Länge k: $\underbrace{E \circ \cdots \circ E}_{k \text{ mal}}$

Nicht Path sondern Walk!

Wege als transitiver Abschluss der Kantenrelation

- Kantenrelation E: Menge von Wegen der Länge 1 zwischen Knoten
- Menge von Wegen der Länge 2 als Komposition von E mit sich selber:
  $E \circ E = \{(v_1, v_2) \in V^2 | \exists u \in V : (v_1, u) \in E \wedge (u, v_2) \in E\}$
- Menge von Wegen einer Länge k: $\underbrace{E \circ \cdots \circ E}_{k \text{ mal}}$
- Transitiver Abschluss als Vereinigung dieser Mengen:
  $E^* = E \cup E \circ E \cup \cdots \cup \underbrace{E \circ \cdots \circ E}_{? \text{ mal}}$

Nicht Path sondern Walk!

Wege als transitiver Abschluss der Kantenrelation

- Kantenrelation E: Menge von Wegen der Länge 1 zwischen Knoten
- Menge von Wegen der Länge 2 als Komposition von E mit sich selber:
  $E \circ E = \{(v_1, v_2) \in V^2 | \exists u \in V : (v_1, u) \in E \wedge (u, v_2) \in E\}$
- Menge von Wegen einer Länge k: $\underbrace{E \circ \cdots \circ E}_{k \text{ mal}}$
- Transitiver Abschluss als Vereinigung dieser Mengen:
  $E^* = E \cup E \circ E \cup \cdots \cup \underbrace{E \circ \cdots \circ E}_{n \text{ mal}}$

Nicht Path sondern Walk!

Potenzieren der Adjazenzmatrix

- Komposition von E als logisches Matrixprodukt der Adjazenzmatrix $\mathbf{M}$

Potenzieren der Adjazenzmatrix

- Komposition von E als logisches Matrixprodukt der Adjazenzmatrix $\mathbf{M}$
- Weitere Komposition durch das Potenzieren von $\mathbf{M}$

Potenzieren der Adjazenzmatrix

- Komposition von E als logisches Matrixprodukt der Adjazenzmatrix $\mathbf{M}$
- Weitere Komposition durch das Potenzieren von $\mathbf{M}$
- Vereinigen der Mengen durch das logische **oder** der Matrizen
- Logisches **oder** zweier Matrizen wird komponentenweise gebildet
  - $(\mathbf{A} \vee \mathbf{B})_{ij} = \mathbf{A}_{ij} \vee \mathbf{B}_{ij}$

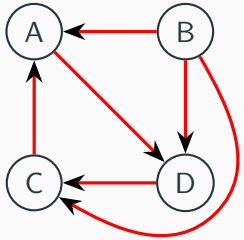
Potenzieren der Adjazenzmatrix

- Komposition von E als logisches Matrixprodukt der Adjazenzmatrix $\mathbf{M}$
- Weitere Komposition durch das Potenzieren von $\mathbf{M}$
- Vereinigen der Mengen durch das logische **oder** der Matrizen
- Logisches **oder** zweier Matrizen wird komponentenweise gebildet
  - $(\mathbf{A} \vee \mathbf{B})_{ij} = \mathbf{A}_{ij} \vee \mathbf{B}_{ij}$

Berechnung der Erreichbarkeitsmatrix $\mathbf{M}^*$

$$\mathbf{M}^* = \mathbf{M} \vee \mathbf{M}^2 \vee \cdots \vee \mathbf{M}^n$$

Beispiel: Potenzieren der Adjazenzmatrix



$$\mathbf{M} = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{M}^2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{M}^3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \qquad \mathbf{M}^4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{M}^* = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Warshall's Algorithmus

Warshall's Algorithmus

- Matrixmultiplikation ist aufwändig
  - Berechnen der Potenzen bei Graphen mit vielen Knoten ist problematisch

Warshall's Algorithmus

- Matrixmultiplikation ist aufwändig
  - Berechnen der Potenzen bei Graphen mit vielen Knoten ist problematisch
- Warshall ermittelt $\mathbf{M}^*$ in Worst-Case $\Theta(n^3)$ Zeit
- Warshall's Algorithmus arbeitet *In-Place*
  - $\mathbf{M}$ wird iterativ in $\mathbf{M}^*$ überführt

Warshall's Algorithmus

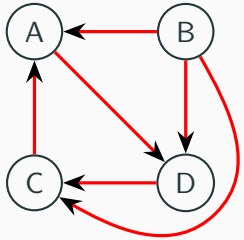
- Matrixmultiplikation ist aufwändig
  - Berechnen der Potenzen bei Graphen mit vielen Knoten ist problematisch
- Warshall ermittelt $\mathbf{M}^*$ in Worst-Case $\Theta(n^3)$ Zeit
- Warshall's Algorithmus arbeitet *In-Place*
  - $\mathbf{M}$ wird iterativ in $\mathbf{M}^*$ überführt
- Warshall erzeugt Matrizen $\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_n$
  - Wobei $\mathbf{W}_0 = \mathbf{M}$ und $\mathbf{W}_n = \mathbf{M}^*$
- $\mathbf{W}_k[i, j] = 1 \iff$ Es existiert ein Weg von v_i nach v_j, wobei alle Zwischenknoten $\in \{v_1, v_2, \ldots, v_k\}$ sind

Pseudocode: Warshall's Algorithmus

Algorithm 1 Warshall's Algorithmus

```
1:  function WARSHALL(M)
2:      W ← M
3:      for k = 1 to n do
4:          for i = 1 to n do
5:              if W[i, k] then
6:                  for j = 1 to n do
7:                      W[i, j] ← W[i, j] ∨ W[k, j]
8:                  end for
9:              end if
10:         end for
11:     end for
12:     return W
13: end function
```

Beispiel: Warshall's Algorithmus

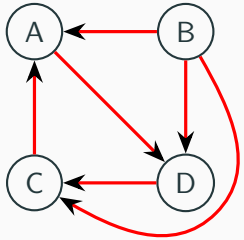


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

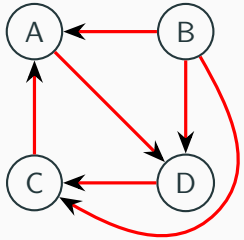


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

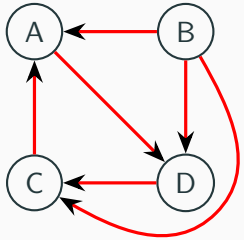


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ & & & \\ & & & \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus



$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

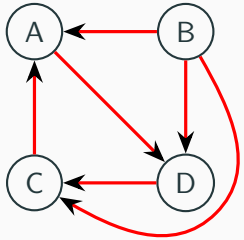
$$\mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ & & & \\ & & & \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

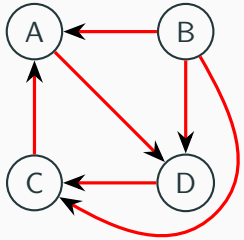


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

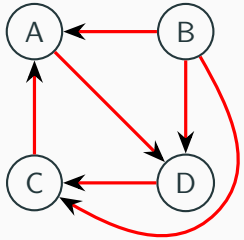


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

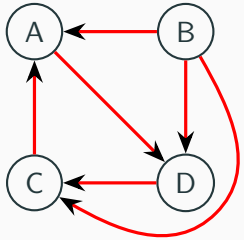


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

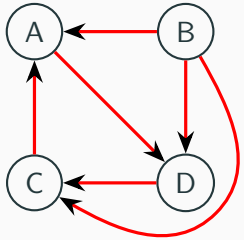


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

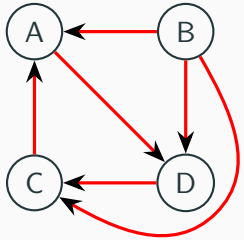


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

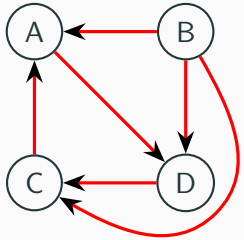


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

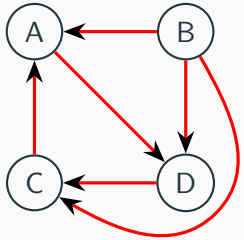


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

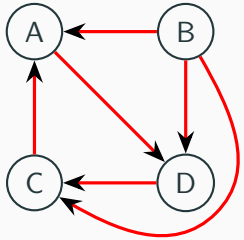


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

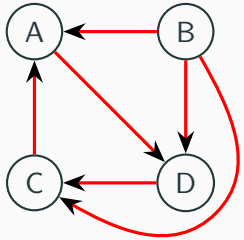


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ & & & \\ 1 & 0 & 0 & 1 \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

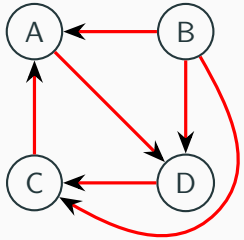


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ & & & \\ 1 & 0 & 0 & 1 \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

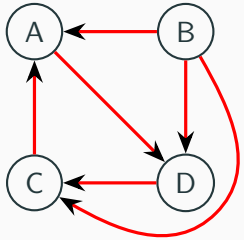


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

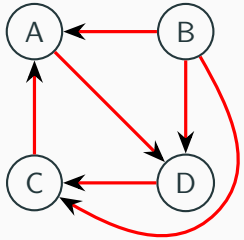


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ & & & \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

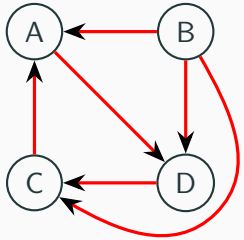


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

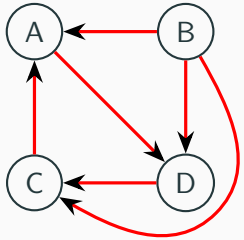


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

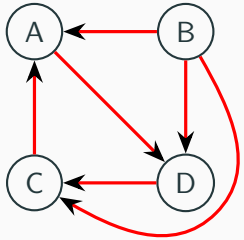


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

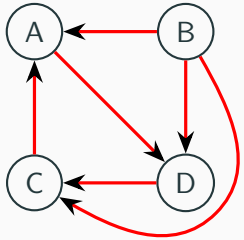


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

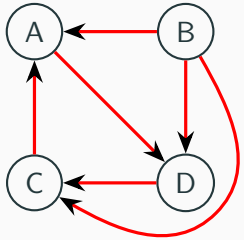


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

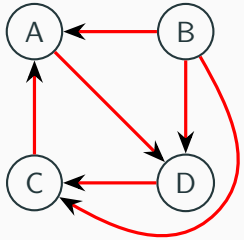


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

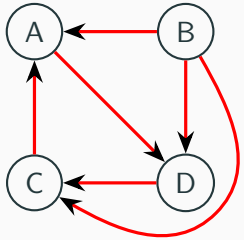


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

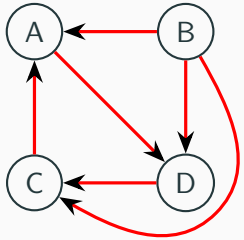


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

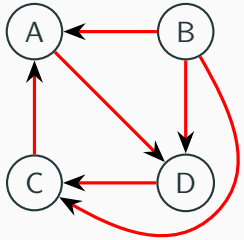


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

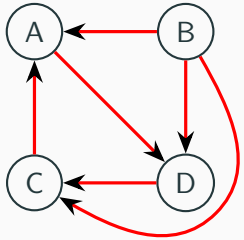


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ & & & \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus

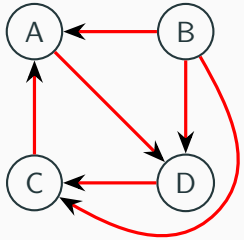


$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Beispiel: Warshall's Algorithmus



$$\mathbf{W}_0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_1 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{W}_2 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \mathbf{W}_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{W}_4 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Kürzeste Wege

Kürzeste Wege

- Gewichteter Digraph: Jede Kante bekommt ein reelle Zahl zugeordnet
  - Gewicht, Kosten, Distanz, . . . dieser Kante

Kürzeste Wege

- Gewichteter Digraph: Jede Kante bekommt ein reelle Zahl zugeordnet
  - Gewicht, Kosten, Distanz, . . . dieser Kante
- Auffindung von Wegen minimaler Gesamtdistanz

Kürzeste Wege

- Gewichteter Digraph: Jede Kante bekommt ein reelle Zahl zugeordnet
  - Gewicht, Kosten, Distanz, . . . dieser Kante
- Auffindung von Wegen minimaler Gesamtdistanz
- Motivation: Routingprotokolle (OSPF), Routenplanung, Basis für Suchalgorithmen aus der künstlichen Intelligenz

Kürzeste Wege

- Gewichteter Digraph: Jede Kante bekommt ein reelle Zahl zugeordnet
  - Gewicht, Kosten, Distanz, . . . dieser Kante
- Auffindung von Wegen minimaler Gesamtdistanz
- Motivation: Routingprotokolle (OSPF), Routenplanung, Basis für Suchalgorithmen aus der künstlichen Intelligenz
- Gewichteter Digraph lässt sich durch Gewichtsmatrix $\mathbf{W}$ repräsentieren, wobei

$$\mathbf{W}_{ij} = \begin{cases} 0, & i = j \\ \infty, & (v_i, v_j) \notin E \\ d(v_i, v_j), & sonst \end{cases}$$

Dijkstra's Algorithmus

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*
  - Findet zu einem gegebenen Startknoten einen kürzesten Weg zu jedem anderen Knoten des Digraphen

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*
  - Findet zu einem gegebenen Startknoten einen kürzesten Weg zu jedem anderen Knoten des Digraphen
  - Eine Modifikation des Algorithmus von Warshall (Floyd-Warshall-Algorithmus) löst das *All-Nodes Shortest-Path Problem*

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*
  - Findet zu einem gegebenen Startknoten einen kürzesten Weg zu jedem anderen Knoten des Digraphen
  - Eine Modifikation des Algorithmus von Warshall (Floyd-Warshall-Algorithmus) löst das *All-Nodes Shortest-Path Problem*
- Vorausetzung für Dijkstra's Algorithmus sind nicht negative Kantengewichte für alle Kanten

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*
  - Findet zu einem gegebenen Startknoten einen kürzesten Weg zu jedem anderen Knoten des Digraphen
  - Eine Modifikation des Algorithmus von Warshall (Floyd-Warshall-Algorithmus) löst das *All-Nodes Shortest-Path Problem*
- Vorausetzung für Dijkstra's Algorithmus sind nicht negative Kantengewichte für alle Kanten
  - Ein alternativer Algorithmus (Bellman-Ford-Algorithmus) löst das Problem auch für negative Kantengewichte (jedoch generell langsamer)

Dijkstra's Algorithmus

- Dijkstra's Algorithmus löst das *Single-Source Shortest-Path Problem*
  - Findet zu einem gegebenen Startknoten einen kürzesten Weg zu jedem anderen Knoten des Digraphen
  - Eine Modifikation des Algorithmus von Warshall (Floyd-Warshall-Algorithmus) löst das *All-Nodes Shortest-Path Problem*
- Vorausetzung für Dijkstra's Algorithmus sind nicht negative Kantengewichte für alle Kanten
  - Ein alternativer Algorithmus (Bellman-Ford-Algorithmus) löst das Problem auch für negative Kantengewichte (jedoch generell langsamer)
  - Gilt jedoch nur, sofern keine negative Zyklen (Zyklen mit negativem Gesamtgewicht) existieren
    - Sonst gibt es keinen eindeutigen Shortest-Path

Dijkstra's Algorithmus

- Dijkstra betrachtet im Algorithmus die (aktuelle) Distanz $d(v)$ eines Knotens v vom Startknoten A

Dijkstra's Algorithmus

- Dijkstra betrachtet im Algorithmus die (aktuelle) Distanz $d(v)$ eines Knotens v vom Startknoten A
- Trivialerweise ist $d(A) = 0$, alle anderen Knoten bekommen zunächst eine Distanz von ∞ zugewiesen (sind also noch nicht erreichbar)

Dijkstra's Algorithmus

- Dijkstra betrachtet im Algorithmus die (aktuelle) Distanz $d(v)$ eines Knotens v vom Startknoten A
- Trivialerweise ist $d(A) = 0$, alle anderen Knoten bekommen zunächst eine Distanz von ∞ zugewiesen (sind also noch nicht erreichbar)
- In jedem Schritt des Algorithmus wird ein Knoten betrachtet, abgearbeitet und *markiert*
  - Markierte Knoten gelten als abgearbeitet und werden nicht weiter betrachtet oder verändert

Dijkstra's Algorithmus

- Es wird stets der von A am nächsten liegende Knoten u (welcher nicht markiert ist) ausgewählt und markiert

Dijkstra's Algorithmus

- Es wird stets der von A am nächsten liegende Knoten u (welcher nicht markiert ist) ausgewählt und markiert
- Von diesem Knoten aus werden alle benachbarte Knoten v (welche nicht markiert sind) betrachtet

Dijkstra's Algorithmus

- Es wird stets der von A am nächsten liegende Knoten u (welcher nicht markiert ist) ausgewählt und markiert
- Von diesem Knoten aus werden alle benachbarte Knoten v (welche nicht markiert sind) betrachtet
  - Es wird die Distanz von A nach v als $d(u) + d(u, v)$ berechnet (die Distanz von A nach u und von u nach v)
  - Ist diese Summe kleiner als die aktuelle Distanz von A nach v, wird $d(v)$ mit der zuvor berechneten Summe aktualisiert

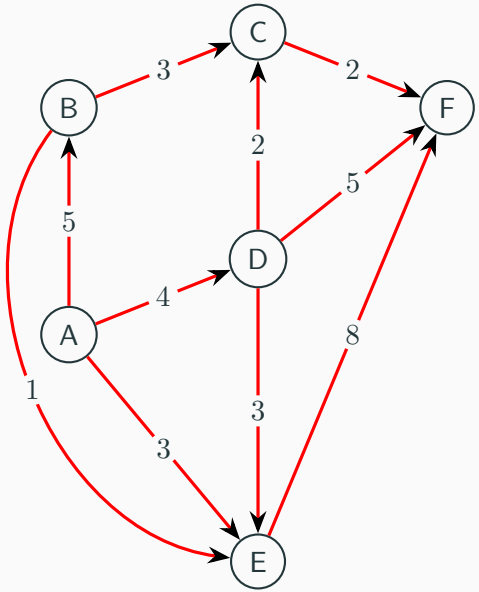
Dijkstra's Algorithmus

- Es wird stets der von A am nächsten liegende Knoten u (welcher nicht markiert ist) ausgewählt und markiert
- Von diesem Knoten aus werden alle benachbarte Knoten v (welche nicht markiert sind) betrachtet
  - Es wird die Distanz von A nach v als $d(u) + d(u, v)$ berechnet (die Distanz von A nach u und von u nach v)
  - Ist diese Summe kleiner als die aktuelle Distanz von A nach v, wird $d(v)$ mit der zuvor berechneten Summe aktualisiert
- Dies wird solange wiederholt, bis alle Knoten einmal betrachtet (und markiert) wurden

Dijkstra's Algorithmus

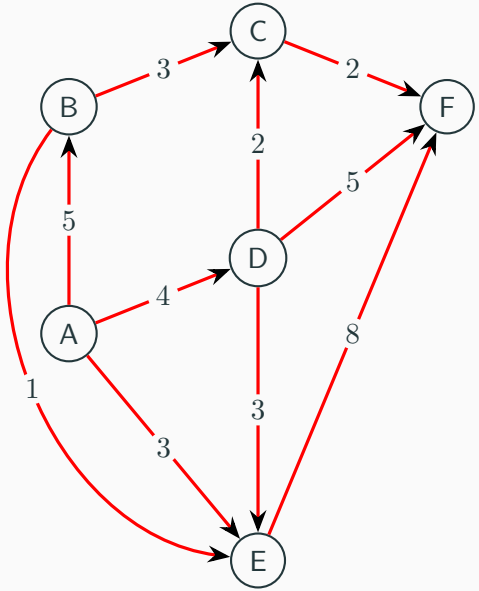
- Um später auch den kürzesten Weg von A zu einem beliebigen Knoten v rekonstruieren zu können, wird üblicherweise auch ein *Parent*-Attribut $p(v)$ mitgeführt
  - Dieses enthält den Vorgänger von v im aktuell kürzesten Weg von A nach v
  - Dieses Feld wird im Laufe des Algorithmus zusammen mit $d(v)$ aktualisiert

Beispiel: Dijkstra's Algorithmus



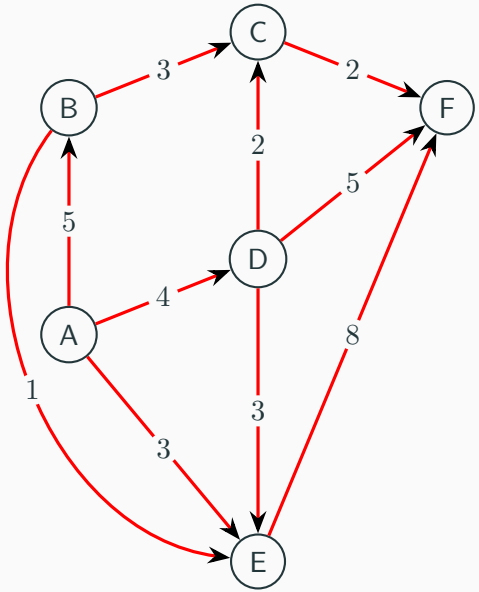
v	$d(v)$	$p(v)$
A		
B		
C		
D		
E		
F		

Beispiel: Dijkstra's Algorithmus



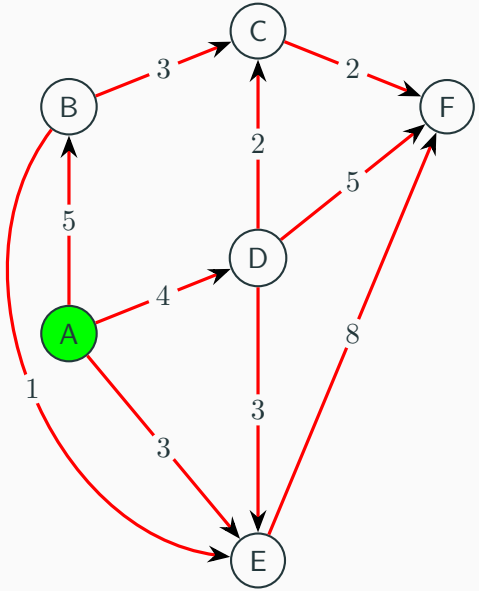
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B		
C		
D		
E		
F		

Beispiel: Dijkstra's Algorithmus



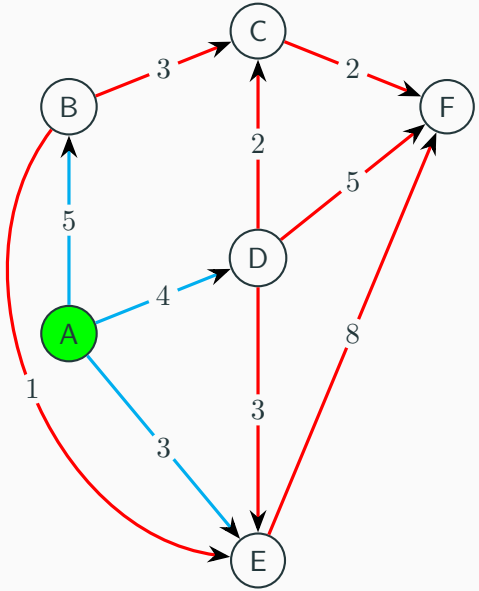
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	∞	$\varnothing$
C	∞	$\varnothing$
D	∞	$\varnothing$
E	∞	$\varnothing$
F	∞	$\varnothing$

Beispiel: Dijkstra's Algorithmus



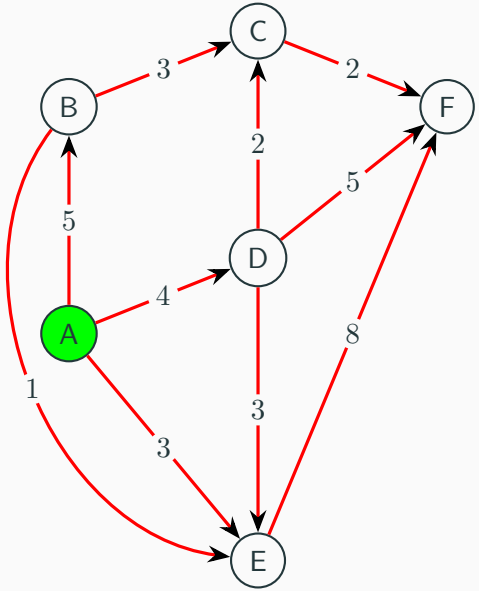
v	$d(v)$	$p(v)$
A	0	∅
B	∞	∅
C	∞	∅
D	∞	∅
E	∞	∅
F	∞	∅

Beispiel: Dijkstra's Algorithmus



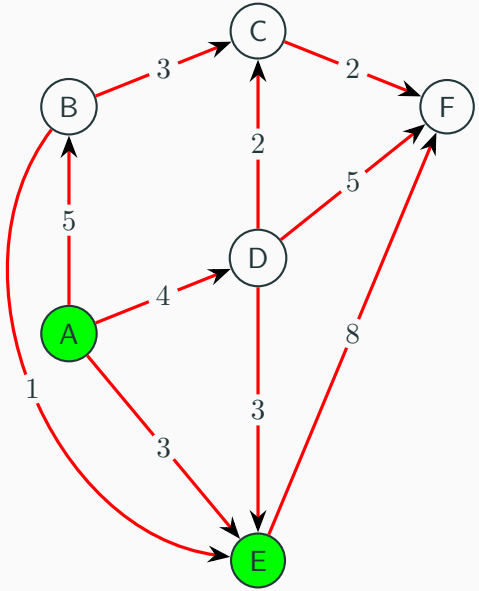
v	$d(v)$	$p(v)$
A	0	∅
B	∞	∅
C	∞	∅
D	∞	∅
E	∞	∅
F	∞	∅

Beispiel: Dijkstra's Algorithmus



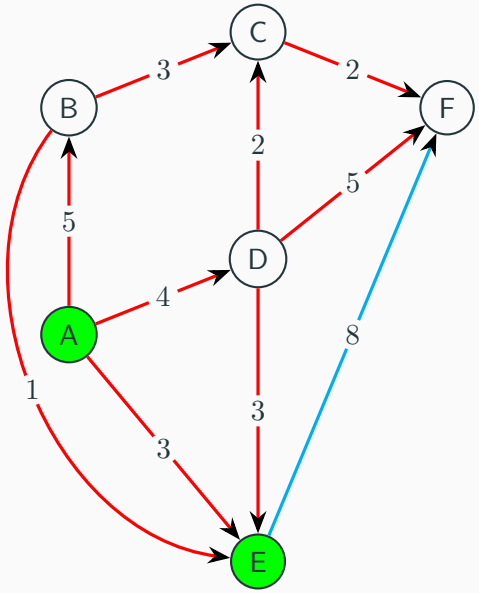
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	∞	∅

Beispiel: Dijkstra's Algorithmus



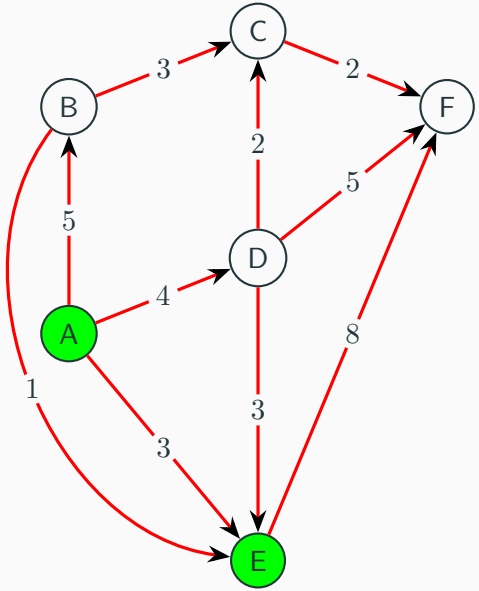
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	∞	∅

Beispiel: Dijkstra's Algorithmus



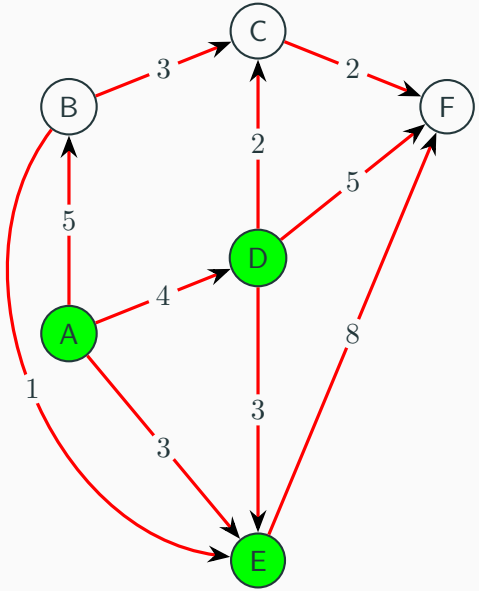
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	∞	∅

Beispiel: Dijkstra's Algorithmus



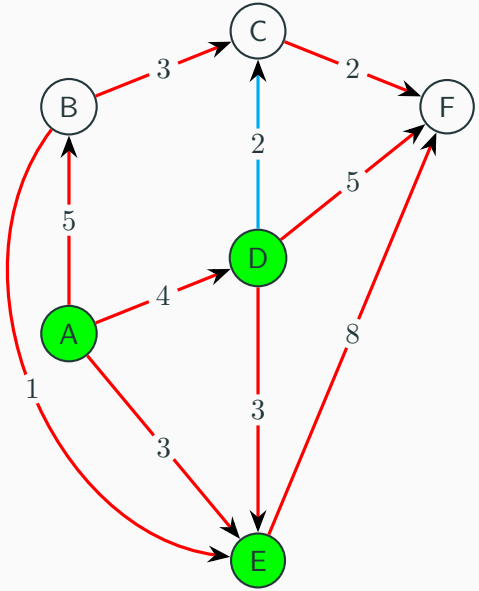
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	11	E

Beispiel: Dijkstra's Algorithmus



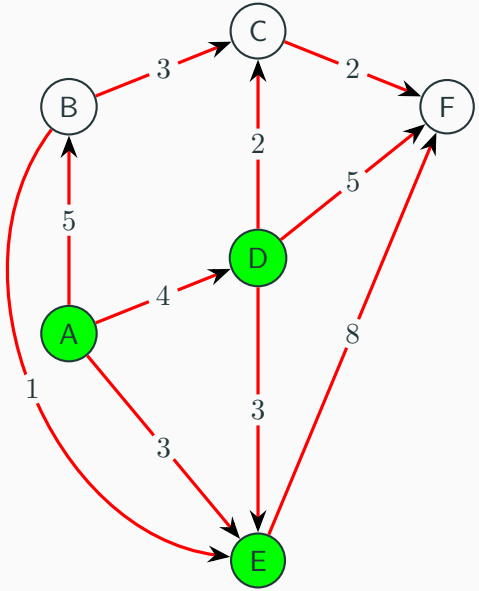
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	11	E

Beispiel: Dijkstra's Algorithmus



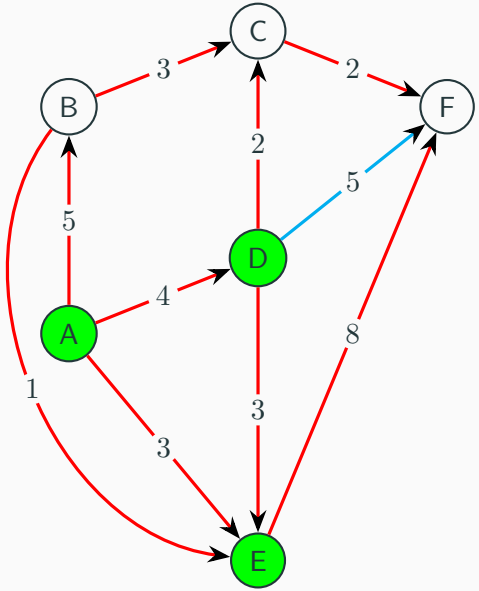
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	∞	∅
D	4	A
E	3	A
F	11	E

Beispiel: Dijkstra's Algorithmus



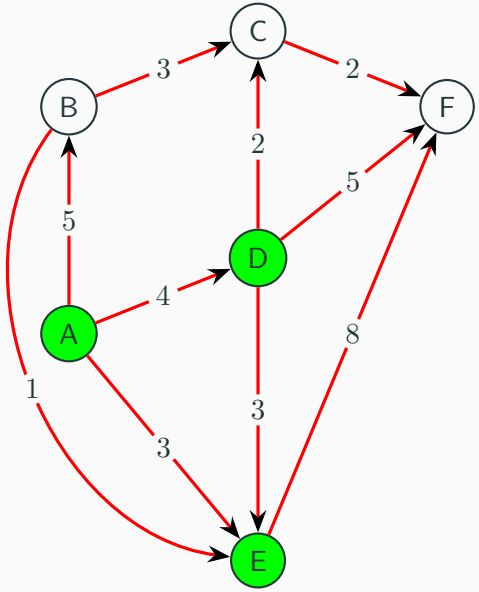
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	6	D
D	4	A
E	3	A
F	11	E

Beispiel: Dijkstra's Algorithmus



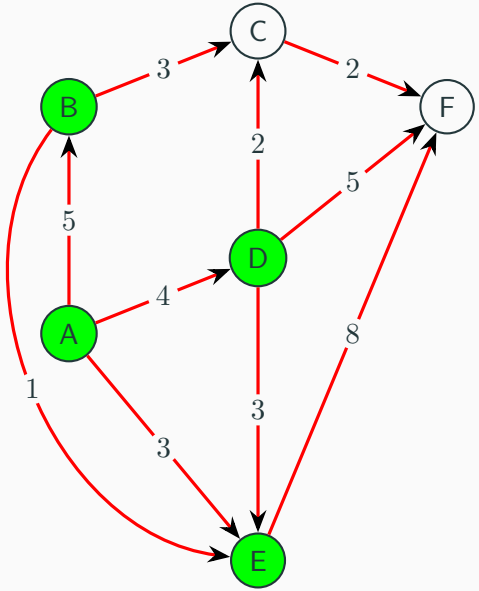
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	11	E

Beispiel: Dijkstra's Algorithmus



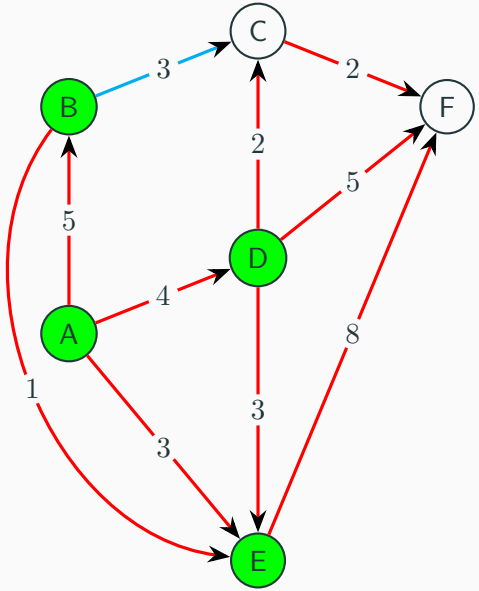
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



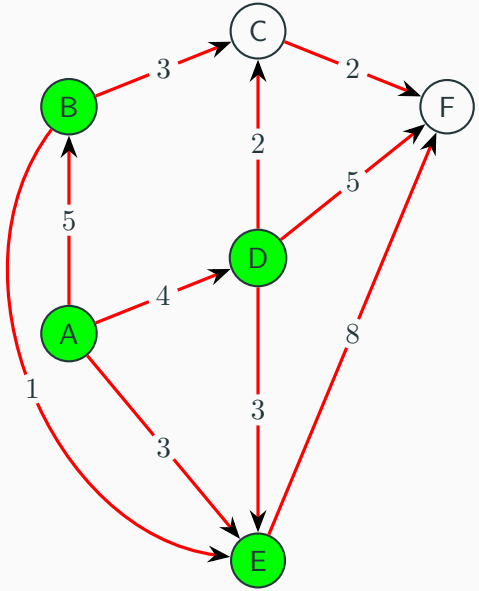
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



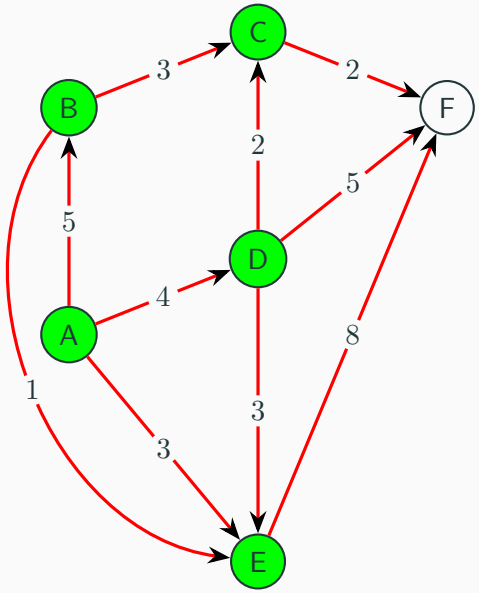
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



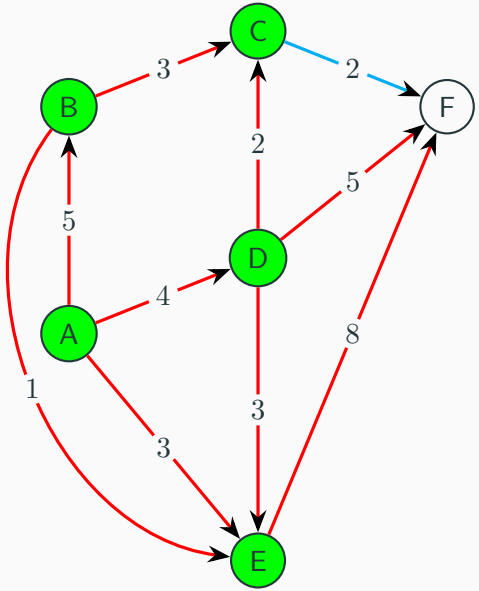
v	$d(v)$	$p(v)$
A	0	∅
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



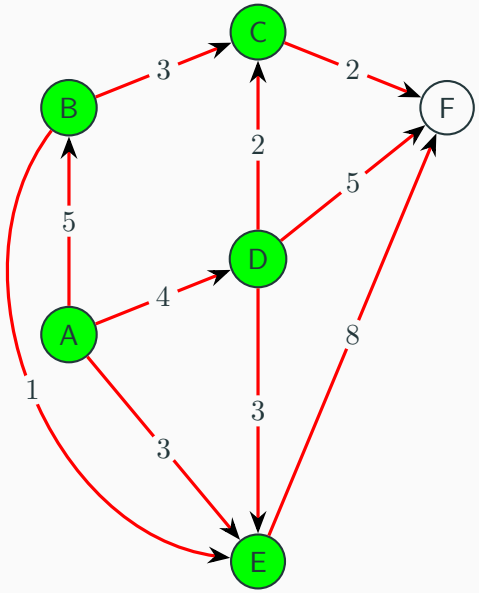
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



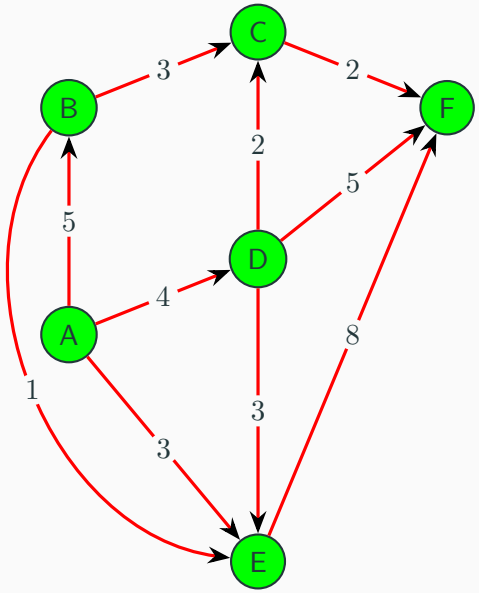
v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	9	D

Beispiel: Dijkstra's Algorithmus



v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	8	C

Beispiel: Dijkstra's Algorithmus

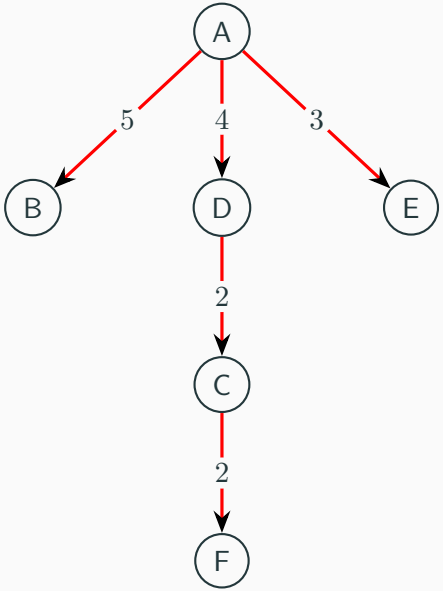


v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	8	C

Shortest-Path Baum

v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	8	C

Shortest-Path Baum



v	$d(v)$	$p(v)$
A	0	$\varnothing$
B	5	A
C	6	D
D	4	A
E	3	A
F	8	C