Westfälische
Hochschule
Gelsenkirchen Bocholt Recklinghausen

Grundlagen der Informatik und Programmierung 2

Ausgewählte Algorithmen auf Graphen

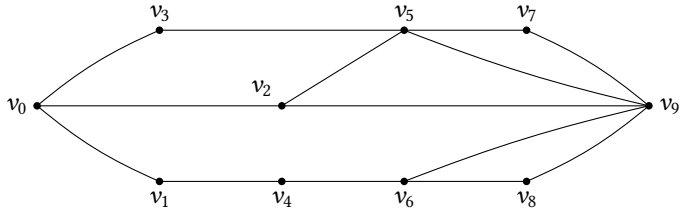
Floyd-Warshall-Algorithmus

Prof. Dr. Tom Vierjahn

Visual Computing (https://vc.w-hs.de)
Fachbereich Wirtschaft und Informationstechnik
Campus Bocholt

Sommersemester 2020

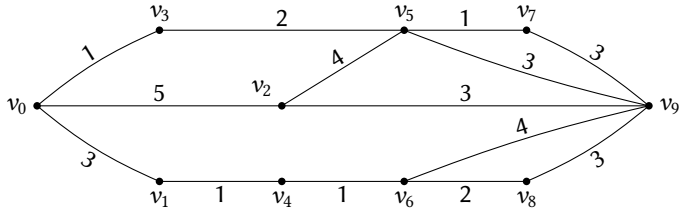
Kantengewichtete Graphen



Definition: Kantengewichteter Graph

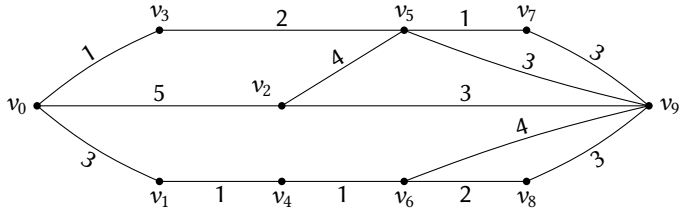
Ein kantengewichteter Graph G ist ein geordnetes Tripel $G = (V, E, f)$, bestehend aus einer Menge V an Knoten (Vertices), einer (Multi-)Menge E an Kanten (Edges) und einer Funktion $f : E \rightarrow \mathbb{R}$, die jeder Kante $e_i \in E$ ein (Kanten-)gewicht $f(e_i) = w_i$ zuordnet.

Adjazenzmatrix im kantengewichteten Graphen

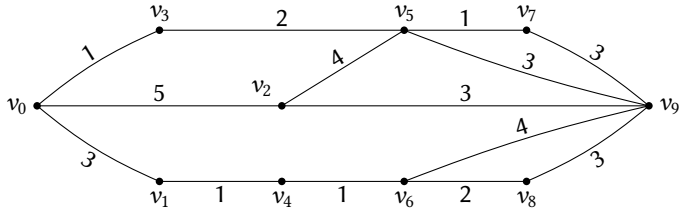


- Die Kantengewichte eines gewichteten Graphen $G = (V, E, f)$ lassen sich z.B. mit der Adjazenzmatrix angeben:

Kürzeste Wege im kantengewichteten Graphen

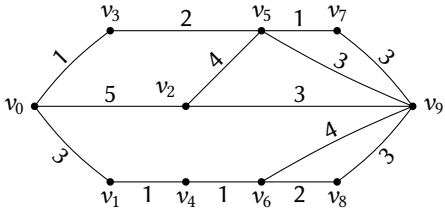


Aufgabe



Floyd-Warshall-Algorithmus[1]

Initialisierung



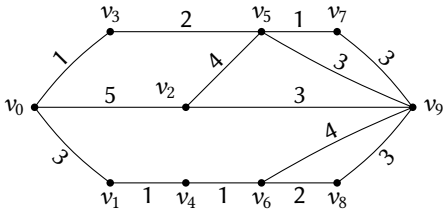
	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0										
v_1										
v_2										
v_3										
v_4										
v_5										
v_6										
v_7										
v_8										
v_9										

- Entfernung jedes Knotens zu sich selbst:
- Entfernung entlang Kanten:
- Übrige Entfernungen:

[1] benannt nach dessen Erfindern Robert Floyd (1936–2001), Stephen Warshall (1935–2006)

Floyd-Warshall-Algorithmus

Schritt 1

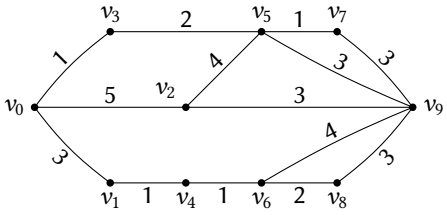


- Wähle Zwischenknoten $v_z = v_0$
- Wähle Startknoten $v_s = v_0$
- Wähle Endknoten $v_e = v_0$
- Distanz $d(v_s, v_z) = 0$
- Distanz $d(v_z, v_e) = 0$
- Update: $d'(v_s, v_e) = \min(d(v_s, v_e), d(v_s, v_z) + d(v_z, v_e))$

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	∞	∞	∞	∞	∞	∞
v_1	3	0	∞	∞	1	∞	∞	∞	∞	∞
v_2	5	∞	0	∞	∞	4	∞	∞	∞	3
v_3	1	∞	∞	0	∞	2	∞	∞	∞	∞
v_4	∞	1	∞	∞	0	∞	1	∞	∞	∞
v_5	∞	∞	4	2	∞	0	∞	1	∞	3
v_6	∞	∞	∞	∞	1	∞	0	∞	2	4
v_7	∞	∞	∞	∞	∞	1	∞	0	∞	3
v_8	∞	∞	∞	∞	∞	∞	2	∞	0	3
v_9	∞	∞	3	∞	∞	3	4	3	3	0

Floyd-Warshall-Algorithmus

Schritt 13

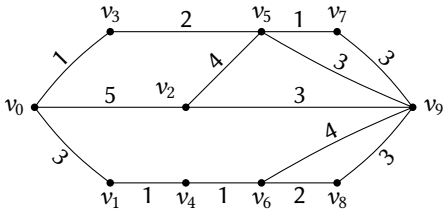


- Wähle Zwischenknoten $v_z = v_0$
- Wähle Startknoten $v_s = v_1$
- Wähle Endknoten $v_e = v_2$
- Distanz $d(v_s, v_z) =$
- Distanz $d(v_z, v_e) =$
- Update: $d'(v_s, v_e) =$
  $\min(d(v_s, v_e), d(v_s, v_z) + d(v_z, v_e))$

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	∞	∞	∞	∞	∞	∞
v_1	3	0		∞	1	∞	∞	∞	∞	∞
v_2	5	∞	0	∞	∞	4	∞	∞	∞	3
v_3	1	∞	∞	0	∞	2	∞	∞	∞	∞
v_4	∞	1	∞	∞	0	∞	1	∞	∞	∞
v_5	∞	∞	4	2	∞	0	∞	1	∞	3
v_6	∞	∞	∞	∞	1	∞	0	∞	2	4
v_7	∞	∞	∞	∞	∞	1	∞	0	∞	3
v_8	∞	∞	∞	∞	∞	∞	2	∞	0	3
v_9	∞	∞	3	∞	∞	3	4	3	3	0

Floyd-Warshall-Algorithmus

Schritt 14

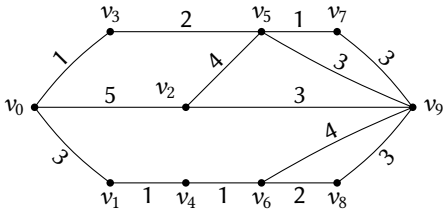


- Wähle Zwischenknoten $v_z = v_0$
- Wähle Startknoten $v_s = v_1$
- Wähle Endknoten $v_e = v_3$
- Distanz $d(v_s, v_z) =$
- Distanz $d(v_z, v_e) =$
- Update: $d'(v_s, v_e) =$
  $\min(d(v_s, v_e), d(v_s, v_z) + d(v_z, v_e))$

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	∞	∞	∞	∞	∞	∞
v_1	3	0	8		1	∞	∞	∞	∞	∞
v_2	5	∞	0	∞	∞	4	∞	∞	∞	3
v_3	1	∞	∞	0	∞	2	∞	∞	∞	∞
v_4	∞	1	∞	∞	0	∞	1	∞	∞	∞
v_5	∞	∞	4	2	∞	0	∞	1	∞	3
v_6	∞	∞	∞	∞	1	∞	0	∞	2	4
v_7	∞	∞	∞	∞	∞	1	∞	0	∞	3
v_8	∞	∞	∞	∞	∞	∞	2	∞	0	3
v_9	∞	∞	3	∞	∞	3	4	3	3	0

Floyd-Warshall-Algorithmus

Schritt 210

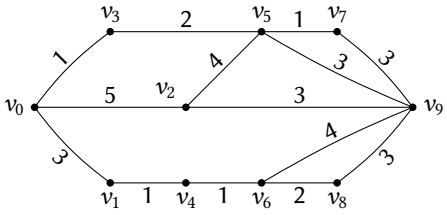


- Wähle Zwischenknoten $v_z = v_2$
- Wähle Startknoten $v_s = v_0$
- Wähle Endknoten $v_e = v_9$
- Distanz $d(v_s, v_z) =$
- Distanz $d(v_z, v_e) =$
- Update: $d'(v_s, v_e) =$
  $\min(d(v_s, v_e), d(v_s, v_z) + d(v_z, v_e))$

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	4	9	∞	∞	∞	
v_1	3	0	8	4	1	∞	∞	∞	∞	∞
v_2	5	8	0	6	9	4	∞	∞	∞	3
v_3	1	4	6	0	5	2	∞	∞	∞	∞
v_4	4	1	9	5	0	∞	1	∞	∞	∞
v_5	∞	∞	4	2	∞	0	∞	1	∞	3
v_6	∞	∞	∞	∞	1	∞	0	∞	2	4
v_7	∞	∞	∞	∞	∞	1	∞	0	∞	3
v_8	∞	∞	∞	∞	∞	∞	2	∞	0	3
v_9	∞	∞	3	∞	∞	3	4	3	3	0

Floyd-Warshall-Algorithmus

Schritt 510

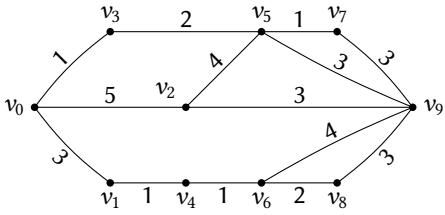


- Wähle Zwischenknoten $v_z = v_5$
- Wähle Startknoten $v_s = v_0$
- Wähle Endknoten $v_e = v_9$
- Distanz $d(v_s, v_z) =$
- Distanz $d(v_z, v_e) =$
- Update: $d'(v_s, v_e) =$
  $\min(d(v_s, v_e), d(v_s, v_z) + d(v_z, v_e))$

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	4	3	5	4	∞	
v_1	3	0	8	4	1	6	2	∞	∞	11
v_2	5	8	0	6	9	4	10	∞	∞	3
v_3	1	4	6	0	5	2	6	∞	∞	9
v_4	4	1	9	5	0	7	1	∞	∞	12
v_5	3	6	4	2	7	0	8	1	∞	3
v_6	5	2	10	6	1	8	0	∞	2	4
v_7	∞	∞	∞	∞	∞	1	∞	0	∞	3
v_8	∞	∞	∞	∞	∞	∞	2	∞	0	3
v_9	8	11	3	9	12	3	4	3	3	0

Floyd-Warshall-Algorithmus

Ergebnis



	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	3	5	1	4	3	5	4	7	6
v_1	3	0	8	4	1	6	2	7	4	6
v_2	5	8	0	6	8	4	7	5	6	3
v_3	1	4	6	0	5	2	6	3	8	5
v_4	4	1	8	5	0	7	1	8	3	5
v_5	3	6	4	2	7	0	7	1	6	3
v_6	5	2	7	6	1	7	0	7	2	4
v_7	4	7	5	3	8	1	7	0	6	3
v_8	7	4	6	8	3	6	2	6	0	3
v_9	6	6	3	5	5	3	4	3	3	0

Floyd-Warshall-Algorithmus[1]

Finde kürzeste Distanzen (und Pfade) zwischen allen Knoten.

Westfälische Hochschule

Initialisierung:

- Initialisiere $|V|$-mal-$|V|$ Distanzenmatrix auf ∞.
- Setze Hauptdiagonale auf 0
- Trage Kantengewichte ein.

Für jedes Tripel aus Zwischen- (v_z), Start- (v_s), Endknoten (v_e):

- Bestimme Distanz von Start- zum Zwischenknoten: $d(v_s, v_z)$
- Bestimme Distanz vom Zwischen- zum Endknoten: $d(v_z, v_e)$
- Aktualisiere Distanz vom Start- zum Zielknoten,
  falls $d(v_s, v_z) + d(v_z, v_e)$ kleiner als ursprüngliche Distanz.
- Falls auch kürzeste Pfade bestimmt werden: aktualisiere Vorgänger in Vorgängermatrix.

[1] benannt nach dessen Erfindern Robert Floyd (1936–2001), Stephen Warshall (1935–2006)

Zusammenfassung

Westfälische Hochschule

- Kantengewichte
- Floyd-Warshall-Algorithmus am Beispiel eines ungerichteten, kantengewichteten Graphen
- kürzeste Entfernungen/Wege

Prof. Dr. Tom Vierjahn

- E-Mail: tom.vierjahn@w-hs.de

Visual Computing

- Web: `https://vc.w-hs.de`
- YouTube: Visual Computing WH
- Twitter: @VisComputingWH

Westfälische Hochschule
Fachbereich Wirtschaft und Informationstechnik
Campus Bocholt