1 Kürzeste Wege

Der vermutlich bekannteste Algorithmus geht auf E.W. Dijkstra (1959) zurück, ein anderer auf Moore-Bellman (1958). Beide bestimmen zu einem Knoten v die kürzesten Wege zu allen anderen Knoten in $\mathcal{O}\left(n^2\right)$, also alle kürzesten Wege in $\mathcal{O}\left(n^3\right)$.

1.1 Floyd-Warshall-Algorithmus (1962)

Algorithmus 1 Floyd-Warshall-Algorithmus

Eingabe: Gerichteter Graph $G = (V, E)$, $V = \{1, 2, \ldots, n\}$ mit Gewichten $c(e)$ für alle Kanten $e \in E$.

Ausgabe: Zwei Matrizen $W = (w_{ij})$ und $P = (p_{ij})$, wobei w_{ij} für $i \neq j$ die Länge eines kürzesten (i, j)-Weges und w_{ii} die Länge eines kürzesten Zykels ist, der i enthält. P_{ij} ist der vorletzte Knoten eines kürzesten (i, j)-Weges (bzw. (i, j)-Zykels).

Schritt 1:

for $i = 1$ bis n **do**
  for $j = 1$ bis n **do**

$$w_{ij} := \begin{cases} c(i,j) & \text{falls } (i,j) \in E \\ \infty & \text{sonst} \end{cases}$$

$$p_{ij} := \begin{cases} i & \text{falls } (i,j) \in E \\ 0 & \text{sonst} \end{cases}$$

  end for
end for

Schritt 2:

for $k = 1$ bis n **do**
  for $i = 1$ bis n **do**
    for $j = 1$ bis n **do**
      if $w_{ij} > w_{ik} + w_{kj}$ **then**
        $w_{ij} := w_{ik} + w_{kj}$
        $p_{ij} := p_{kj}$
      end if
    end for
  end for
end for

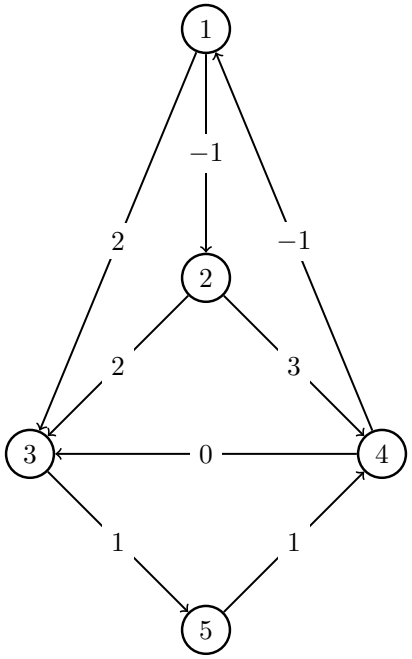
Schritt 3: Ausgabe von W und P.

Bemerkung Falls in Schritt 2 $i = j$ und $w_{ii} < 0$, so kann abgebrochen werden - ein „kürzester“ Zykel ist in diesem Fall nicht wohldefiniert.

W^0 bezeichne die Anfangsmatrix, W^k die Matrix, die im k-ten Durchlauf der äußeren Schleife von Schritt 2 berechnet wird. Dabei berechent sich $W^k = \left(w_{ij}^k\right)$ aus W^{k-1} für $k = 1, 2, \ldots, n$ folgendermaßen:

$$w_{ij}^k := min\left\{w_{ij}^{k-1}, w_{ik}^{k-1} + w_{kj}^{k-1}\right\}$$

Beispiel 1.1 Ein gerichter Graph G mit $n = |V(G)| = 5$ Knoten und $|E(G)| = 8$ Kanten. $V(G) = \{1, 2, 3, 4, 5\}$, $E(G) = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 5), (4, 1), (4, 3), (5, 4)\}$, sowie Kantengewichten.



$$W^0 = \left(\begin{array}{c|cccc} \infty & -1 & 2 & \infty & \infty \\ \hline \infty & \infty & 2 & 3 & \infty \\ \infty & \infty & \infty & \infty & 1 \\ -1 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & \infty \end{array}\right) \quad W^1 = \left(\begin{array}{cc|ccc} \infty & -1 & 2 & \infty & \infty \\ \hline \infty & \infty & 2 & 3 & \infty \\ \hline \infty & \infty & \infty & \infty & 1 \\ -1 & -2 & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & \infty \end{array}\right)$$

$$W^2 = \left(\begin{array}{cc|cc|c} \infty & -1 & 1 & 2 & \infty \\ \infty & \infty & 2 & 3 & \infty \\ \hline \infty & \infty & \infty & \infty & 1 \\ \hline -1 & -2 & 0 & 1 & \infty \\ \infty & \infty & \infty & 1 & \infty \end{array}\right) \quad W^3 = \left(\begin{array}{cc|cc|c} \infty & -1 & 1 & 2 & 2 \\ \infty & \infty & 2 & 3 & 3 \\ \infty & \infty & \infty & \infty & 1 \\ \hline 1 & -2 & 0 & 1 & 1 \\ \hline \infty & \infty & \infty & 1 & \infty \end{array}\right)$$

$$W^4 = \left(\begin{array}{cccc|c} 1 & -1 & 1 & 2 & 2 \\ 2 & 1 & 2 & 3 & 3 \\ \infty & \infty & \infty & \infty & 1 \\ -1 & -2 & 0 & 1 & 1 \\ \hline 0 & -1 & 1 & 1 & 2 \end{array}\right) \qquad \boldsymbol{W^5} = \left(\begin{array}{ccccc} \mathbf{1} & \mathbf{-1} & \mathbf{1} & \mathbf{2} & \mathbf{2} \\ \mathbf{2} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{3} \\ \mathbf{1} & \mathbf{0} & \mathbf{2} & \mathbf{2} & \mathbf{1} \\ \mathbf{-1} & \mathbf{-2} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{-1} & \mathbf{1} & \mathbf{1} & \mathbf{2} \end{array}\right)$$

Satz 1.1 Sei G ein gerichteter Graph mit Kantengewichten $c(e)$ für alle $e \in E$. Sei W die $n \times n$-Matrix, die von Floyd-Warhall-Algorithmus berechnet wird. Dann gilt

a) Der Floyd-Warshall-Algorithmus liefert genau dann eine Kürzeste-Weglängen Matrix W, wenn G keine negativen Kreise(Zykel) enthält.

b) G enthält genau dann einen negativen Kreis, wenn ein Hauptdiagonal-element von G negativ ist.

BEWEIS Induktion über k.

Zeige: Besitzt G *a*) einen bzw. *b*) keinen negativen Kreis in der Knotenmenge $\{1, 2, \ldots, k\}$, so ist

1. $w_{ii}^k < 0$ für ein $i \in \{1, 2, \ldots, k\}$
2. w^k die Matrix der kürzesten Längen von (ij)-Wegen bzw (i, i)-Kreisen, bei denen nur die Knoten $1, 2, \ldots, k$ als innere Knoten auftreten können

Induktionsanfang: $k = 0$ offensichtlich Schritt 1

Induktionsschritt: $k \to k + 1$

Angenommen die Behauptung gilt für ein $k \geq 0$ und Schritt 2 ist zum $(k + 1)$-ten Male durchlaufen. Dabei wird

$$w_{ij}^{k+1} := \min\left\{w_{ij}^k, w_{ik+1}^k + w_{k+1j}^k\right\}$$

gesetzt. Das heißt w_{ij}^{k+1} ist die Länge eines kürzesten (i, j)-Weges, der nur Knoten aus $\{1, 2, \ldots, k + 1\}$ enthält, es sei denn $w_{ij}^k > w_{ik+1}^k + w_{k+1j}^k$ und die Verkettung K des $(i, k + 1)$-Weges mit $(k + 1, j)$-Weg enthält Zykel in K. Durch Wegnahme aller Zykel in K erhalten wir einen (i, j)-Weg $\overline{K}$. Nach Konstruktion ist der Knoten $k + 1$ in einem der Zykel enthalten, also ist $\overline{K}$ ein (i, j)-Weg, der nur Knoten aus $\{1, 2, \ldots, k\}$ enthält. Daher folgt: $w_{ij}^k \leq c\left(\overline{K}\right)$. Da aber $c(K) = w_{ik+1}^k + w_{k+1j}^k < w_{ij} \leq c\left(\overline{K}\right)$ gilt, hat mindestens ein Kreis in K negative Länge. Für jeden Knoten i aus diesem negativen Kreis ist daher $w_{ii}^{k+1} < 0$

Der Floyd-Warshall-Algorithmus hat eine Laufzeit von $\mathcal{O}\left(n^3\right)$. ∎

Bemerkung Für einen gerichteten Graphen G, der keine negativen Kreise enthält, kann ein kürzester gerichteter Zykel in $\mathcal{O}\left(n^3\right)$ Schritten bestimmt werden.

Bei einer ganzen Reihe von Problemen in der Graphentheorie und in der Praxis taucht die Bestimmung aller kürzesten Wege als (Teil-)Problem auf.

Eine verbreitete Anwendung findet das kürzeste-Wege-Problem bei der Erstellung einer Distanzmatrix (z.B. Autobahnen).

Wenn lediglich die kürzesten Wege von einem Knoten x_0 zu allen anderen Knoten aus G interessant sind, kann auch der folgende Algorithmus nach Dijkstra verwendet werden:

1.2 Dijkstra-Algorithmus

Algorithmus 2 Dijkstra-Algorithmus

Eingabe: gewichteter Graph $G = (V, E)$, Gewichtsfunktion $c : E \to \mathbb{R}^+$, Startknoten x_0

Ausgabe:

- Kantenmenge U mit den Kanten der kürzesten Wege von x_0 zu allen anderen Knoten
- Funktion $t : V \to \mathbb{R}$ mit den Entfernungen von x_0 zu $y \in V$.

Schritt 1:

$$t(x) := \begin{cases} 0 & x = x_0 \\ \infty & x \in V \setminus \{x_0\}, S := \{x_0\}, U := \emptyset \end{cases}$$

Schritt 2:

while $\overline{S} \neq \emptyset$ **do**
wähle eine Kante $x^*y^* \in E$ mit $x^* \in S$ und $y^* \in \overline{S}$ und $t(x^*) + c(x^*y^*)$ minimal
$t(y^*) := t(x^*) + c(x^*y^*)$
$S := S \cup \{y^*\}$, $U := U \cup \{x^*y^*\}$
end while

Satz 1.2 Sei $G = (V, E)$ ein gewichteter Graph. Dann berechnet der Dijkstra-Algorithmus alle kürzesten Wege von x_0 zum Rest der Knoten.

Beweis Wir zeigen die Aussage mittels Induktion bezüglich der Schleifendurchläufe in Schritt 2.
Falls $V = \{x_0\}$, so ist die Aussage trivialerweise erfüllt. Für den Induktionsschritt genügt es nur zu zeigen, dass für jeden Knoten v, der zu S hinzugefügt wird, $t(v) \leq d(x_0, v)$ gilt. Angenommen $t(v) > d(x_0, v)$. Sei $P = x_0 \dots x_k[= v]$ ein kürzester Weg von x_0 nach v der Länge $d(x_0, v)$. Sei $i \in \{0, \dots, k\}$ minimal mit $x_i \in V \setminus S$. Dann gilt $i \geq 1$, da $x_0 \in S$ bereits vor der ersten Iteration gilt und niemals Knoten aus S entfernt werden.

Es ist $x_{i-1} \in S$ nach Wahl von i. Dann wurde x_{i-1} in einer früheren Iteration als Minimum in S eingefügt. Nach Induktionsvorraussetzung galt daher $t(x_{i-1}) \leq d(x_0, x_{i-1})$. Dann gilt aber

$$t(x_i) \leq t(x_{i-1}) + c(x_{i-1}, x_i) = d(x_0, x_{i-1}) + c(x_{i-1}, x_i) = d(x_0, x_i)$$

Also ist $t(x_i) \leq d(x_o, x_i) \leq d(x_0, v) \leq t(v)$, was ein Widerspruch zur Wahl von v als Knoten für S darstellt. ∎