

## 6. Kürzeste Wege

Gegeben sind ein (Di)Graph  $G = (V, E)$  und eine Gewichtsfunktion  $w : E \rightarrow \mathbb{R}^+ \cup \{+\infty\}$ . O. B. d. A. sei  $G$  vollständig, damit auch zusammenhängend.

Sei  $u = v_0, v_1, v_2, \dots, v_n = v$  ein Pfad in  $G$ . Die Länge dieses Pfades ist

$$\sum_{i=0}^{n-1} w(v_i, v_{i+1}).$$

$d(u, v)$  sei die Länge eines kürzesten Pfades von  $u$  nach  $v$ .

## Problemstellungen:

- 1 Gegeben  $u, v \in V$ , berechne  $d(u, v)$ .
- 2 Gegeben  $u \in V$ , berechne für alle  $v \in V$  die Länge  $d(u, v)$  eines kürzesten Pfades von  $u$  nach  $v$  (**sssd, single source shortest distance**).
- 3 Berechne für alle  $(u, v) \in V^2$  die kürzeste Entfernung  $d(u, v)$  (**apsd, all pairs shortest distance**).
- 4 Die Probleme **sssp, single source shortest path** und **apsp, all pairs shortest path** sind entsprechend, nur wird jeweils ein kürzester Pfad (und nicht nur dessen Länge) berechnet.

## 6.1 Der Floyd-Warshall-Algorithmus für apsd

Gegeben sind ein (Di)Graph  $G = (V, E)$  und eine Gewichtsfunktion  $w : E \rightarrow \mathbb{R}^+ \cup \{+\infty\}$ . Sei o. B. d. A.  $V = \{0, \dots, n-1\}$ . Eine Gewichtsmatrix ist wie folgt definiert:

$$D = (w(v_i, v_j))_{\substack{0 \leq i < n \\ 0 \leq j < n}}$$

Ziel ist es, eine  $n \times n$ -Matrix mit den Einträgen

$$d_{ij} = \text{Länge eines kürzesten Weges von } i \text{ nach } j$$

zu berechnen. Dazu werden induktiv Matrizen  $D^{(k)}$  mit Einträgen

$$d_{ij}^{(k)} = \left( \begin{array}{l} \text{Länge eines kürzesten Weges von } i \text{ nach } j, \\ \text{so dass alle inneren Knoten } < k \text{ sind} \end{array} \right)$$

erzeugt.

algorithm Floyd

for  $i=0$  to  $n-1$  do

for  $j=0$  to  $n-1$  do

$D^0[i, j] := w(v_i, v_j)$

od

od

for  $k=0$  to  $n-1$  do

for  $i=0$  to  $n-1$  do

for  $j=0$  to  $n-1$  do

$D^{k+1}[i, j] := \min\{D^k[i, j],$   
 $D^k[i, k] + D^k[k, j]\}$

od

od

od

end

## Satz 315

*Der Floyd-Algorithmus berechnet für alle  $u, v \in V^2$  die Länge eines kürzesten Weges zwischen  $u$  und  $v$ , und zwar mit Zeitbedarf  $\Theta(n^3)$  und Platzbedarf  $\Theta(n^2)$ .*

**Beweis:**

Ersichtlich aus Algorithmus.



## Bemerkungen:

- 1 Zur Bestimmung der eigentlichen Pfade (und nicht nur der Entfernungen) muss bei der Minimum-Bestimmung jeweils das  $k$  gespeichert werden.
- 2 Der Algorithmus funktioniert auch, wenn *negative Kantengewichte* vorhanden sind, es jedoch keine *negativen Kreise* gibt.
- 3 Die Erweiterung auf Digraphen ist offensichtlich.

## 6.2 Dijkstras Algorithmus für sssd

Gegeben sind ein (Di)Graph  $G = (V, E)$ , ein Knoten  $s \in V$  und eine Gewichtsfunktion  $w : E \rightarrow \mathbb{R}^+ \cup \{\infty\}$ .

algorithm Dijkstra

$F := V \setminus \{s\}$

for all  $v \in F$  do  $d[v] := w(s, v)$  od

co  $d[s] = 0$  oc

while  $F \neq \emptyset$  do

bestimme  $v \in F$  mit  $d[v]$  minimal

$F := F \setminus \{v\}$

for all  $w \in N(v)$  do

$d[w] := \min\{d[w], d[v] + w(v, w)\}$

od

od

end

## Satz 316

*Dijkstras Algorithmus berechnet  $d(s, v)$  für alle  $v \in V$ ; der Zeitaufwand ist  $O(n^2)$ , der Platzbedarf  $O(n + m)$ .*

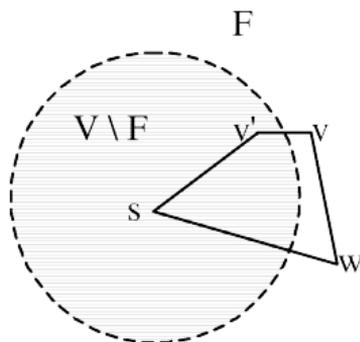
### Beweis:

Zeit- und Platzbedarf sind aus dem Algorithmus ersichtlich. Die Korrektheit zeigen wir mit einem Widerspruchsbeweis:

Annahme:  $v$  sei der erste Knoten, so dass  $d(s, v)$  falsch (d. h. zu groß) berechnet wird.

## Beweis (Forts.):

Diese Situation illustriert folgendes Bild:



Nach Annahme muss dann gelten:

$$d(w) + w(w, v) < d(s, v') + w(v', v) = d(v) .$$

Damit wäre  $d(w)$  aber kleiner als  $d(v)$ , und der Algorithmus hätte  $w$  und nicht  $v$  gewählt. □

### Bemerkung:

Mit besseren Datenstrukturen (**priority queues** – z. B. **Fibonacci heaps**) kann Dijkstras Algorithmus so implementiert werden, dass er z. B. in Zeit  $O(m + n \cdot \log n)$  läuft.

## 7. Matchings

### Definition 317

Sei  $G = (V, E)$  ein Graph.

- 1  $M \subseteq E$  heißt **Matching**, falls alle Kanten in  $M$  paarweise disjunkt sind.
- 2  $M$  heißt **maximales Matching**, falls es kein Matching  $M'$  in  $G$  gibt mit  $M \subsetneq M'$ .
- 3  $M$  heißt **Matching maximaler Kardinalität** (aka **Maximum Matching**), falls es in  $G$  kein Matching  $M'$  mit  $|M'| > |M|$  gibt.
- 4  $m(G)$  ist die Kardinalität eines Maximum Matchings in  $G$ .

## Beispiel 318

 maximales Matching  
(aber nicht Maximum)

 Maximum-Matching  
(natürlich auch maximal)

## 7.1 Matchings in bipartiten Graphen

### Satz 319 („Heiratssatz“)

Sei  $G = (U, V, E)$  ein bipartiter Graph. Dann ist  $m(G) = |U|$  genau dann, wenn gilt:

$$(\forall A \subseteq U) [|A| \leq |N(A)|]$$

Beweis:

„ $\Rightarrow$ “

Offensichtlich.

„ $\Leftarrow$ “

Sei  $M$  ein Maximum Matching in  $G$ .

Annahme: Ein Knoten  $u = u_0 \in U$  sei in  $M$  ungematcht.

Wir beginnen in  $u_0$  eine BFS, wobei wir in den ungeraden Schichten (also von  $U$  aus) nur ungematchte und in den geraden Schichten (also von  $V$  aus) nur gematchte Kanten verwenden. Querkanten bleiben außer Betracht.

**Fall 1:** Die BFS findet in  $V$  einen ungematchten Knoten  $v$ . Dann stoppen wir.

**Fall 2:** Nach Vollendung einer geraden Schicht (mit gematchten Kanten) sind alle Blätter des BFS-Baums gematcht. Seien  $U'$  (bzw.  $V'$ ) die Knoten des aktuellen BFS-Baums in  $U$  (bzw.  $V$ ). Gemäß Annahme ist  $|U'| > |V'|$ , die alternierende BFS kann also fortgesetzt werden. Da  $G$  endlich ist, muss schließlich Fall 1 eintreten.

## Beweis (Forts.):

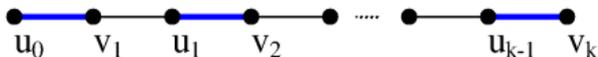
„ $\Leftarrow$ “ (Fortsetzung)

Also existiert per Konstruktion ein Pfad wie in folgender Abbildung:



Ein solcher Pfad, bei dem sich gematchte und ungematchte Kanten abwechseln, heißt **alternierender** Pfad. Sind, wie hier, Anfangs- und Endknoten ungematcht, heißt der Pfad auch **augmentierend**.

Vertauscht man auf diesem Pfad gematchte und ungematchte Kanten, erhält man dadurch ein Matching  $M'$  mit  $|M'| = |M| + 1$ , was wiederum einen Widerspruch darstellt:



## Definition 320

Man definiert für einen bipartiten Graphen  $G = (U, V, E)$  die Kenngröße:

$$\delta := \delta(G) := \max_{A \subseteq U} \{|A| - |N(A)|\}$$

Da bei der Maximumsbildung auch  $A = \emptyset$  sein kann, ist  $\delta \geq 0$ .

## Satz 321

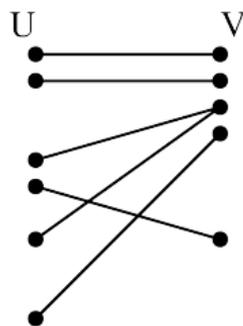
*Es gilt:*

$$m(G) = |U| - \delta .$$

## Beweis:

Dass  $m(G) \leq |U| - \delta$  gilt, ist offensichtlich. Wir zeigen nun noch, dass auch  $m(G) \geq |U| - \delta$  gilt, damit ist der Satz bewiesen.

Betrachte folgenden Graphen:



Man fügt nun  $\delta$  neue Knoten hinzu. Von diesen gehen Kanten zu allen Knoten in  $U$ , so dass ein  $K_{|U|,\delta}$  entsteht.

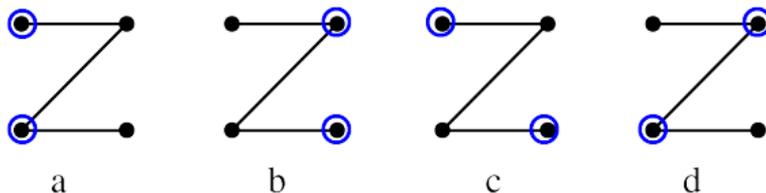
### Beweis (Forts.):

Der neue Graph erfüllt die Voraussetzungen des Heiratssatzes. Damit gibt es im neuen Graphen ein Matching  $M'$  mit  $|M'| = |U|$ . Daraus folgt, dass es im alten Graphen ein Matching der Kardinalität  $\geq |U| - \delta$  geben muss.  $\square$

## Definition 322

$D \subseteq U \uplus V$  heißt **Träger** oder **Knotenüberdeckung** (*vertex cover, VC*) von  $G$ , wenn jede Kante in  $G$  zu mindestens einem  $u \in D$  inzident ist.

## Beispiel 323



In den Fällen a, b und d sind Träger gezeigt, in c nicht.

## Satz 324

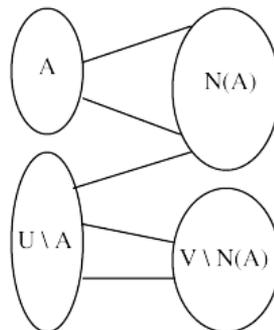
*Es gilt:*

$$\max\{|M|; M \text{ Matching}\} = \min\{|D|; D \text{ Träger}\}$$

## Beweis:

„ $\leq$ “ Offensichtlich.

„ $\geq$ “ Für ein geeignetes  $A \subseteq U$  gilt  $m(G) = |U| - \delta(G) = |U \setminus A| + |N(A)|$ :



$(U \setminus A) \cup N(A)$  ist Träger von  $G$ .



Sei

$$M = (m_{ij})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}}$$

eine (quadratische) Matrix mit  $m_{ij} \geq 0$ . Alle Zeilen- und Spaltensummen von  $M$  seien gleich  $r > 0$ .

Man ordnet nun  $M$  den bipartiten Graphen  $G = (U, V, E)$  zu mit

$$U = \{u_1, \dots, u_n\}, V = \{v_1, \dots, v_n\} \text{ und } \{u_i, v_j\} \in E \Leftrightarrow m_{ij} > 0.$$

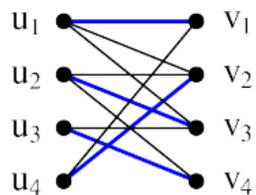
Ein Matching in  $G$  entspricht einer Menge von Positionen in  $M$ , die alle in verschiedenen Zeilen und Spalten liegen.

## Beispiel 325

Die Matrix

$$\begin{pmatrix} \boxed{3} & 1 & 1 & 0 \\ 0 & 1 & \boxed{2} & 2 \\ 0 & 0 & 2 & \boxed{3} \\ 2 & \boxed{3} & 0 & 0 \end{pmatrix}$$

entspricht dem Graphen



### Bemerkung:

Ein Träger  $D$  von  $G$  ist also eine Menge von Zeilen und Spalten von  $M$ , die zusammen alle Einträge  $m_{ij} > 0$  enthalten.

### Definition 326

Eine Menge von Positionen (in  $M$ ), die alle in verschiedenen Zeilen und in verschiedenen Spalten liegen und Einträge  $> 0$  enthalten, heißt **Diagonale** von  $M$ .

Eine Diagonale der Größe  $n$  muss in  $M$  existieren, denn falls  $M$  keine solche Diagonale hat, gibt es nach Satz 324  $e$  Zeilen und  $f$  Spalten mit  $e + f < n$ , die zusammen alle Einträge  $> 0$  von  $M$  enthalten.

Die Gesamtsumme der Einträge in  $M$  wäre dann

$$n \cdot r = \sum_{i,j} m_{ij} \leq (e + f) \cdot r < r \cdot n,$$

was offensichtlich ein Widerspruch ist.