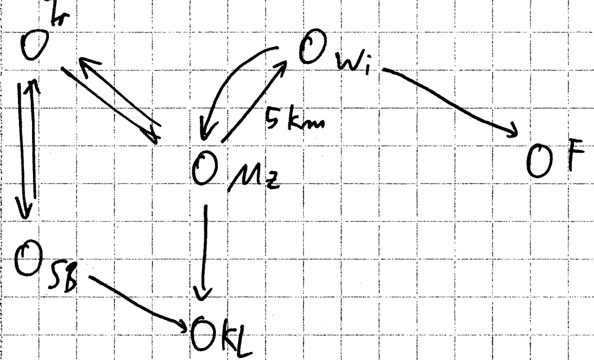
2012-12-03

DSEA.1

Floyd-Warshall-Algorithmus

- Geg.: Graph $G=(V,E)$ $\quad E \subseteq V \times V$ $\quad$ E: Edges, V: Vertices

  $w: E \to \mathbb{R}$ Gewichtsfunktion



Pfad p: $v_0 \to v_1 \to v_i \to v_{i+1} \to v_k$

$v_i \in V$, $(v_i, v_{i+1}) \in E$

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

- Gesucht: Für jedes Knotenpaar ist der kürzeste Weg zwischen diesen beiden Knoten gesucht.

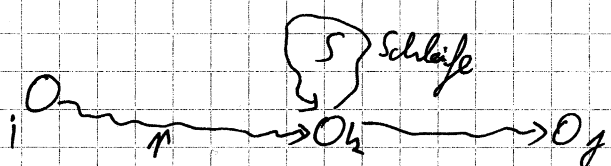
„All-Pair-Shortest-Path Problem"

Datenstruktur $D \in \mathbb{R}^{n \times n}$ $\quad$ Adjazenzmatrix / Abstandsmatrix
(Binärmatrix / Validity) / (gewichtete Werte)

	1	2	...	n
1	0			∞
2		0		
i			$\ddots$	
n				0

$D = (w_{ij})$ $\quad$ $w_{ij} = w(i \to j)$

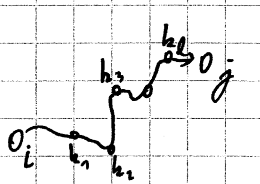
- Beobachtung:



$w(S) < 0 \Rightarrow$ Kürzester Weg nicht wohldef.

$w(S) \geq 0 \Rightarrow$ S kann aus p entfernt werden.

Idee: Optimale Gesamtlösung aus optimalen Teillösungen konstruieren!



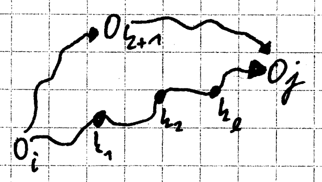
Alle Zwischenknoten sollen einen Index $\leq k$ haben.

$k_1, k_2, k_3, \ldots, k_l \leq k$

2012-12-03

DSEA.2

$d_{ij}^{(k)}$ = Länge eines kürzesten Weges von $i \to j$ sein, mit Zwischenknoten $\leq k$.



$$d_{ij}^{(k+1)} = \min \left\{ d_{ij}^{(k)},\ d_{i,k+1}^{(k)} + d_{k+1,j}^{(k)} \right\}$$

Initialisierung:

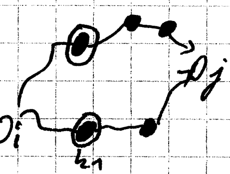
```
for (i = 1; i ≤ n; i++)
    for (j = 1; j ≤ n; ++j)
        d[0][i][j] = w[i][j];
```

```
for (k = 0, k ≤ n, ++k)
    for (i = 1, i ≤ n, ++i)
        for (j = 1, j ≤ n, ++j)
            d[k+1][i][j] = min(d[k][i][j], d[k][i][k+1] + d[k][k+1][j]);
```

Platzsparender: d[i][j] statt d[k][i][j].

ÜB
Warum klappt das?

Weg merken: Merke den ersten Knoten nach i.



Initialisierung von succ:

```
if (w[i][j] < ∞) succ[i][j] = j
else succ[i][j] = -1;
```

```
⇒ for ...
     for ...
        for ...
          if (d[i][j] > d[i][k+1] + d[k+1][j]) {
              d[i][j] = d[i][k+1] + d[k+1][j];   // Weg über k+1
              succ[i][j] = succ[i][k+1];   // Nachfolger: 1. Knoten auf den Weg
                                           //             nach k+1
          }
```

Beobachtung: Sei $v_0 \to v_1 \to \dots v_i \to \dots \to v_j \to \dots \to v_k$ ein kürzester Weg von $v_0 \to v_k$

dann ist $v_i \to v_j$ auch ein kürzester Weg.

12-12-03

DSEA 3

Cocke – Younger – Kasami

$G = (N, \Sigma, S, P)$ kontextfreie Grammatik

Beispiel: Dyckwörter = korrekt geklammerte Wörter

$\Sigma = \{ '(', ')' \}$ Alphabet (terminal Symbole)

$P = \{ S \underset{(1)}{\rightarrow} SS, S \underset{(2)}{\rightarrow} (S), S \underset{(3)}{\rightarrow} \varepsilon \}$ Produktionsregeln

$N = \{ S \}$ Nichtterminal Symbole

S Startsymbol

$$S \underset{(2)}{\rightarrow} (S) \underset{(1)}{\rightarrow} (SS) \underset{(2)}{\rightarrow} ((S)S) \underset{(2)}{\rightarrow} ((S)(S)) \underset{(3)}{\rightarrow} (()(S)) \underset{(3)}{\rightarrow} (()())$$

Wortproblem: Gegeben ein Wort $w = w_1 \cdots w_n \in \Sigma^*$

Frage: kann w mittels der Produktionen aus dem Startsymbol erzeugt werden?

Kontextfreie Grammatiken: $P \subseteq N \times (N \cup \Sigma)^*$

zB $N \rightarrow N_1 N_2 \sigma_1 N_3 \sigma_2$

Nur ein Nichtterminal

Chomsky-Normalform: $N \rightarrow N_1 N_2$, $N \rightarrow \sigma$

$w_1 w_2 w_3 \ldots \underbrace{w_i \ldots w_{i+j-1}} \ldots w_n$

Idee: V_{ij} = Menge aller Nichtterminale, aus der man das Teilwort $w_i \ldots w_{i+j-1}$ ableiten kann.

```
for (i=1; i≤n; ++i)
    if (N→w_i ∈P)
        V_{i,1} = V_{i,1} ∪ {N};
```

Initialisierung $j = 1$.
Teilprobleme der Länge 1.

```
for (j=2; j≤n; ++j)
  for (i=1; i≤n-j+1; ++i)
    for (k=1; k<j; ++k)
      if (N→N_1N_2 ∈P
          ∧ N_1 ∈ V_ik
          ∧ N_2 ∈ V_{i+k, j-k} )
        V_ij = V_ij ∪ {N};
```

$$\underbrace{w_i \ldots w_{i+k-1}}\underbrace{w_{i+k} \ldots w_{i+j-1}}$$

$N \rightarrow N_1 N_2 \in P$ $\quad N_1 \in V_{ik}$ $\quad N_2 \in V_{i+k, j-k}$

$\Rightarrow N \in V_{ij}$