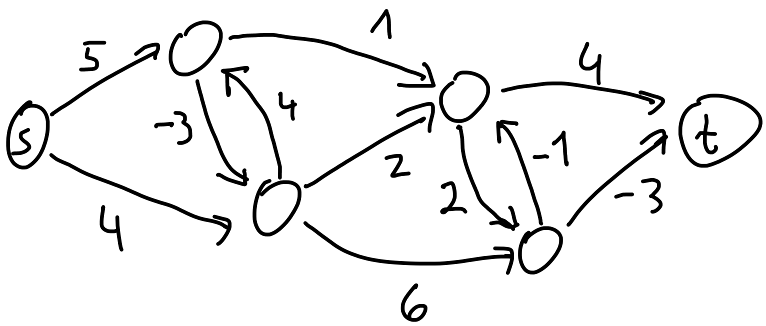
Kürzeste Wege mit negativen Kanten, Algorithmus von Dantzig

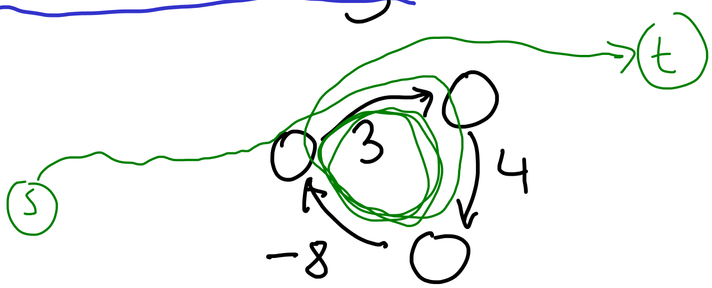


$V = \{v_1, \dots, v_n\}$

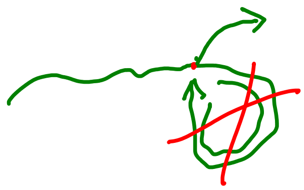
$c_{uv} \in \mathbb{R} \quad (u,v) \in E$

$c_{uv} < 0$ ist erlaubt.

Voraussetzung: keine Kreise negativer Länge!



$\Rightarrow$ kürzeste Wege sind einfache Wege (ohne Knotenwiederholungen)

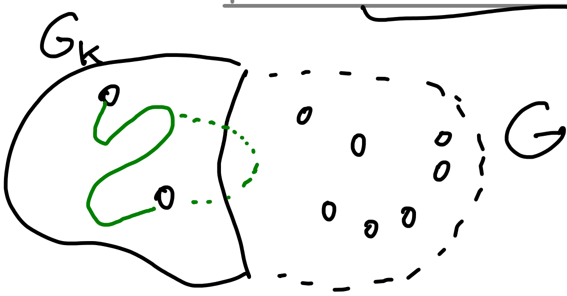


Andernfalls: kürzester Weg undefiniert.

Wir wollen kürzeste Weglängen d_{uv} zwischen allen Paaren $u,v \in V$

Algorithmus von Dantzig

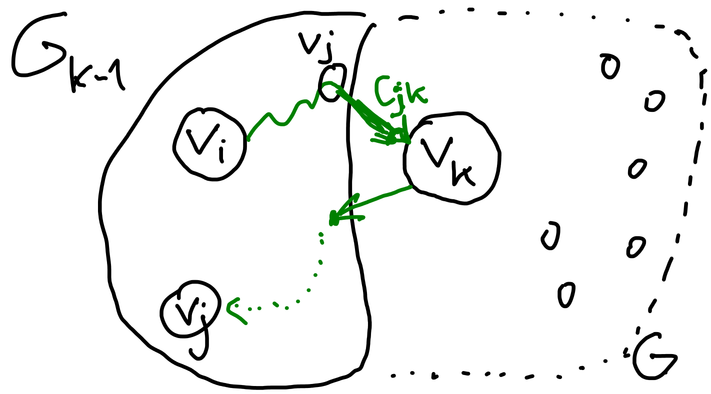
1. $d_{ij}^k = \underbrace{\boxed{\text{kürzeste Weglänge von } v_i \text{ nach } v_j \text{ im Graphen } \{v_1, v_2, \dots, v_k\}}}_{G_k}$ $\qquad k = 1, \dots, n \qquad 1 \le i,j \le k$



$d_{ij}^n = d_{ij}$ = ursprüngliches Problem

2. Übergang $k-1 \to k$

$\boxed{c_{ij} = \infty \text{ falls } (i,j) \notin E}$

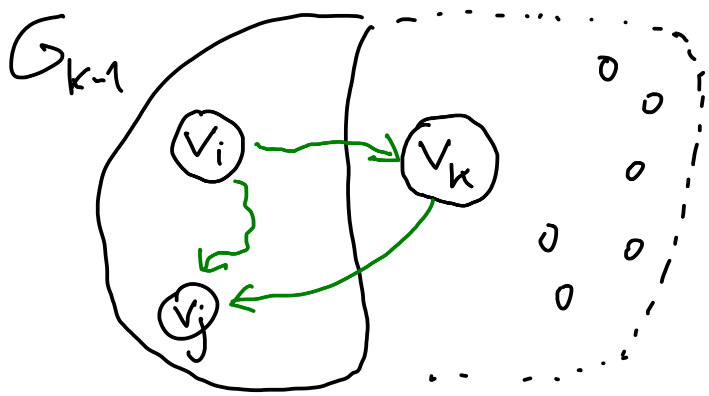


$$d_{ik}^{k} = \min_{j=1,\dots,k-1} \left(d_{ij}^{k-1} + c_{jk} \right)$$

$(i = 1, \dots, k-1)$

$$d_{kj}^{k} = \min_{i=1,\dots,k-1} \left(c_{ki} + d_{ij}^{k-1} \right)$$

$(j = 1, \dots, k-1)$



$$d_{ij}^{k} = \min \{ d_{ik}^{k} + d_{kj}^{k}, d_{ij}^{k-1} \}$$

$(1 \le i,j \le k-1)$

$$d_{kk}^{k} = 0 \quad (*) \leftarrow \text{TEST}$$

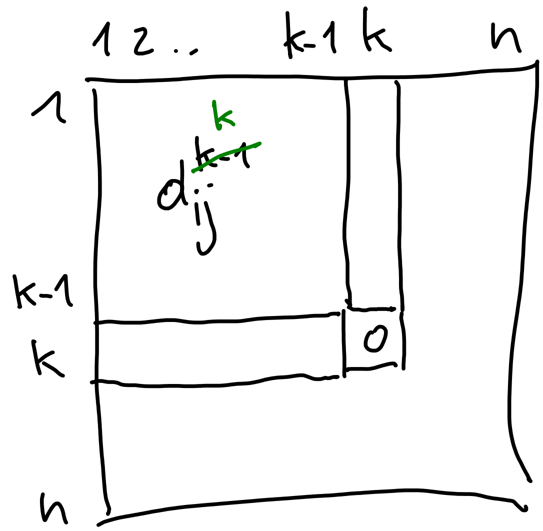
Laufzeit $O(k^2)$

2b $d_{11}^{1} = 0$

....

Gesamtlaufzeit : $O(n^3)$

Speicher : $O(n^2)$

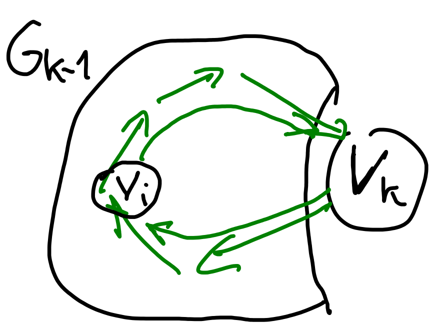


Finden der kürzesten Wege:

Speichere zu jedem i,j einen Zwischenknoten (sofern vorhanden) auf dem kürzesten Weg von i nach j.

Test auf ~~Kreise~~ Kreise negativer Länge

Nehmen wir an, $\underline{G_k}$ ist der erste Graph mit einem negativen Kreis.



$(*)$ Wenn $\min_{i=1,\dots,k-1} \left(d_{ki}^{k} + d_{ik}^{k} \right) < 0$

dann gibt es einen negativen Kreis.

Andere Ansätze:

Algorithmus von Bellman / Ford

fester Startknoten s

$O(mn)$ Zeit

$O(m+n)$ Speicher

$d_j^k :=$ kürzester Abstand von s nach v_j mit höchstens k Kanten

Algorithmus von Floyd / Warshall

alle Knotenpaare

$O(n^3)$ Zeit

$O(n^2)$ Speicher

$d_{ij}^k :=$ kürzester Abstand von v_i nach v_j, der nur die Knoten $\{v_1, v_2, \ldots, v_k\}$ als Zwischenknoten verwendet

$(i, j = 1, 2, \ldots, n)$