Einführung in Operations Research

Flussprobleme (4)

Prof. Dr. Thomas Slawig

CAU Kiel
Institut für Informatik

Inhalt

Lernziele

- Den Algorithmus von Dinic erklären ...
- ... und am Beispiel durchführen können.
- Dazu Details von Schritt (2) – Konstruktion eines zulässigen Flusses, der in jedem (s-t)-Weg eine Kante sättigt.
- Die Aussage über die Komplexität des Algorithmus von Dinic und die Beweisidee dazu kennen ...
- ... und sie im Vergleich zu den bisherigen Algorithmen einordnen können.

Inhalt

Der Algorithmus von Dinic

Ziel: Berechnung eines maximalen Flusses.

Gegeben: zulässiger (s-t)-Fluss in (V, E).

(1) Konstruktion eines Hilfsgraphen (V, E_f) (**Schichtgraph**).
   - (1.1) Bestimme alle **nützlichen** Kanten, d. h. (i, j), wenn $x_{ij} < c_{ij}$ und (j, i), wenn $x_{ij} > \ell_{ij}$.
   - (1.2) Bestimme im **Graphen der nützlichen Kanten** die **Entfernungen** u_j (d. h. Anzahl der Kanten im kürzesten Weg von s zu j) für alle $j = 1, \ldots, n$.
   - (1.3) Berechne **Schichten** $V_\ell := \{j \in V : u_j = \ell\}$ für alle $\ell = 0, 1, \ldots$
   - (1.4) $E_f :=$ alle nützlichen Kanten zwischen zwei aufeinanderfolgenden Schichten $V_\ell \to V_{\ell+1}$.
   - (1.5) Ist t im Schichtgraf von s erreichbar: Setze im Schichtgraph

$$
\begin{aligned}
\bar{\ell}_{ij} &= 0, \\
\bar{c}_{ij} &= c_{ij} - x_{ij}, \ \text{wenn } x_{ij} < c_{ij}, \\
\bar{c}_{ji} &= x_{ij} - \ell_{ij}, \ \text{wenn } x_{ij} > \ell_{ij}.
\end{aligned}
$$

     Sonst: Breche ab (Fluss f ist maximal).

Der Algorithmus von Dinic (Forts.)

(2) Finde im Schichtgraf (V, E_f) einen zulässigen Fluss $\bar{f}$, der in jedem $(s\text{-}t)$-Weg eine Kante sättigt, d. h. für jeden Weg W in (V, E_f) existiert eine Kante auf W mit $\bar{x}_{ij} = \bar{c}_{ij}$.
(Beachte: $\bar{f}$ ist im allgemeinen kein maximaler Fluss.)
Dies kann auf „beliebige“ Art geschehen, z. B. mit dem Algorithmus von Ford-Fulkerson.
Im Original-Algorithmus von Dinic wird eine effizientere Art verwendet, s. u.

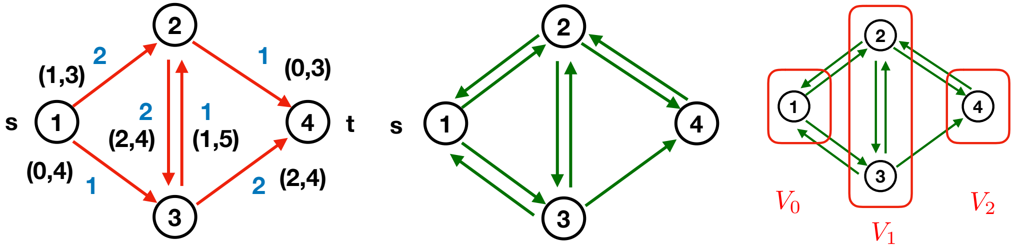
(3) Erhöhe f um $\bar{f}$, d. h. setze

$$x_{ij} := \begin{cases} x_{ij} + \bar{x}_{ij}, & (i,j) \text{ ist Vorwärtskante auf } W, \\ x_{ij} - \bar{x}_{ij}, & (i,j) \text{ ist Rückwärtskante auf } W. \end{cases} \tag{1}$$

$\rightsquigarrow$ Gehe zu (1) (nächste Phase des Algorithmus)

Inhalt

Algorithmus von Dinic: Beispiel



(1) Konstruktion des Schichtgraphen (V, E_f).

(1.1) Bestimme alle **nützlichen** Kanten, d. h. (i, j), wenn $x_{ij} < c_{ij}$ und (j, i), wenn $x_{ij} > \ell_{ij}$.

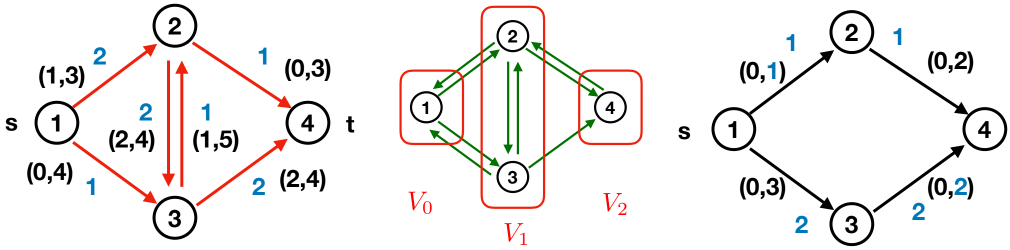
(1.2) Entfernungen u_j im Graphen der nützlichen Kanten:

$$u_1 = 0, u_2 = 1, u_3 = 1, u_4 = 2.$$

(1.3) Berechne Schichten $V_\ell := \{j \in V : u_j = \ell\}$ für alle $\ell = 0, 1, \ldots$:

$$V_0 = \{1\}, V_1 = \{2, 3\}, V_2 = \{4\}.$$

Algorithmus von Dinic: Beispiel



(1) Konstruktion des Schichtgraphen (V, E_f).

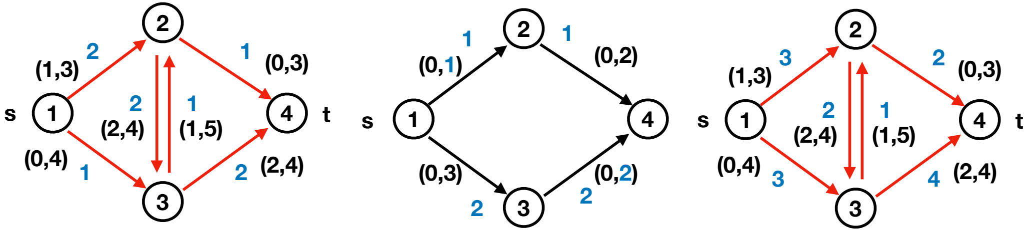
(1.4) E_f: alle nützlichen Kanten zwischen aufeinanderfolgenden Schichten (d. h. $V_\ell \to V_{\ell+1}$).

(1.5) Setze im Schichtgraph

$$\bar{\ell}_{ij} = 0, \quad \bar{c}_{ij} = c_{ij} - x_{ij}, \text{ wenn } x_{ij} < c_{ij}, \quad \bar{c}_{ji} = x_{ij} - \ell_{ij}, \text{ wenn } x_{ij} > \ell_{ij}.$$

(2) Finde im Schichtgraf (V, E_f) zulässigen Fluss $\bar{f}$, der in jedem $(s\text{-}t)$-Weg eine Kante sättigt $(\bar{x}_{ij} = \bar{c}_{ij})$.

Algorithmus von Dinic: Beispiel

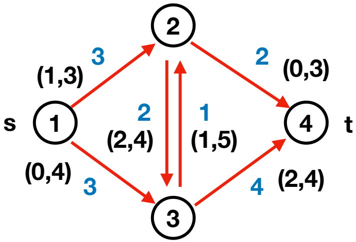


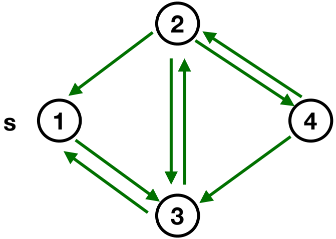
(3) Erhöhe f um $\bar{f}$, d. h. setze

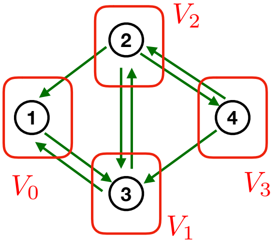
$$x_{ij} := \begin{cases} x_{ij} + \bar{x}_{ij}, & (i,j) \text{ ist Vorwärtskante auf } W, \\ x_{ij} - \bar{x}_{ij}, & (i,j) \text{ ist Rückwärtskante auf } W. \end{cases}$$

$\rightsquigarrow$ Ende 1. Iteration (Phase).

Algorithmus von Dinic: Beispiel







2. Iteration (Phase):

(1) Konstruktion des Schichtgraphen (V, E_f).

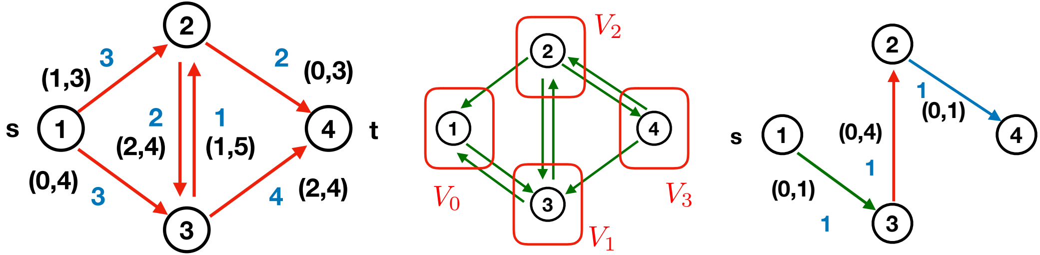
(1.1) Bestimme alle **nützlichen** Kanten, d. h. (i, j), wenn $x_{ij} < c_{ij}$ und (j, i), wenn $x_{ij} > \ell_{ij}$.

(1.2) Entfernungen u_j (Anzahl Kanten im kürzesten Weg $s \to j$): $u_1 = 0, u_2 = 2, u_3 = 1, u_4 = 3$.

(1.3) Schichten $V_\ell := \{j \in V : u_j = \ell\}$

$$V_0 = \{1\}, V_1 = \{3\}, V_2 = \{2\}, V_3 = \{4\}.$$

Algorithmus von Dinic: Beispiel



(1) Konstruktion des Schichtgraphen (V, E_f).

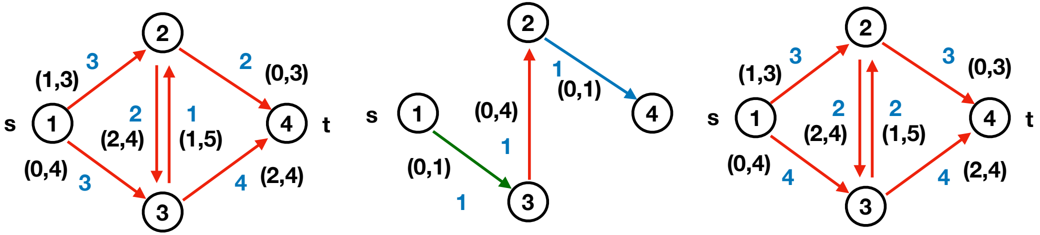
(1.4) E_f: nützlichen Kanten zwischen zwei aufeinanderfolgenden Schichten (V_ℓ nach $V_{\ell+1}$).

(1.5) Ist im Schichtgraf t von s erreichbar? ✓
Setze im Schichtgraph

$$\bar{\ell}_{ij} = 0, \quad \bar{c}_{ij} = c_{ij} - x_{ij}, \text{ wenn } x_{ij} < c_{ij}, \quad \bar{c}_{ji} = x_{ij} - \ell_{ij}, \text{ wenn } x_{ij} > \ell_{ij}.$$

(2) Finde im Schichtgraf zulässigen Fluss $\bar{f}$, der in jedem $(s\text{-}t)$-Weg eine Kante sättigt.

Algorithmus von Dinic: Beispiel

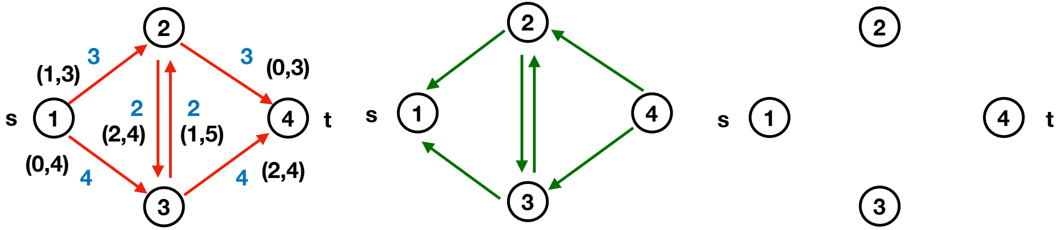


(3) Erhöhe f um $\bar{f}$, d. h. setze

$$x_{ij} := \begin{cases} x_{ij} + \bar{x}_{ij}, & (i,j) \text{ ist Vorwärtskante auf } W, \\ x_{ij} - \bar{x}_{ij}, & (i,j) \text{ ist Rückwärtskante auf } W. \end{cases}$$

$\rightsquigarrow$ Ende 2. Iteration (Phase)

Algorithmus von Dinic: Beispiel



3. Iteration (Phase):

(1) Konstruktion des Schichtgraphen (V, E_f).

(1.1) Bestimme alle **nützlichen** Kanten, d. h. (i, j), wenn $x_{ij} < c_{ij}$ und (j, i), wenn $x_{ij} > \ell_{ij}$.

(1.2) Entfernungen u_j: Hier $u_1 = 0, j = 2, 3, 4$ von $s = 1$ nicht erreichbar.

(1.3) Berechne Schichten $V_\ell := \{j \in V : u_j = \ell\}$: Hier $V_0 = \{1\}$ (einzige nichtleere).

(1.4) E_f: nützlichen Kanten zwischen zwei aufeinanderfolgenden Schichten: keine.

(1.5) $t = 4$ von $s = 1$ nicht erreichbar $\rightsquigarrow$ Abbruch.

Inhalt

Details zu Schritt (2) des Algorithmus

(2) Finde im Schichtgraf (V, E_f) zulässigen Fluss, der in jedem $(s\text{-}t)$-Weg eine Kante sättigt.

(2.1) Berechne Gesamtfluss möglich in Knoten j hinein bzw. aus Knoten j heraus:

$$\textbf{In-Potenzial} \quad P_{in}(j) := \sum_{(i,j)\in E_f} (\bar{c}_{ij} - x_{ij}), \quad P_{in}(s) := \infty,$$

$$\textbf{Out-Potenzial} \quad P_{out}(j) := \sum_{(j,i)\in E_f} (\bar{c}_{ji} - x_{ji}), \quad P_{out}(t) := \infty,$$

$$P(j) := \min\{P_{in}(j), P_{out}(j)\}.$$

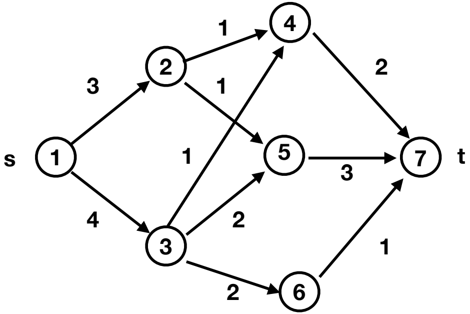
(2.2) Wähle einen Knoten j mit kleinstem Potenzial $P(j)$.
Schicke Fluss der Stärke $P(j)$ von s nach j und von j nach t.
Arbeite Kanten dabei von j aus nacheinander ab und versuche Kanten zu sättigen.
Berechne dabei neue Potenziale und Kapazitäten $\bar{c}_{ij}$.

(2.3) Falls $P(j) = 0$, entferne ein- und ausgehende Kanten von Knoten j aus Schichtgraf (V, E_f).

(2.4) Existiert noch ein $(s\text{-}t)$-Weg in (V, E_f), gehe zu (2.2).

(2.5) Berechne Gesamtfluss als Summe aller so konstruierten Flüsse.

Algorithmus von Dinic: Details zu Schritt (2) – Beispiel



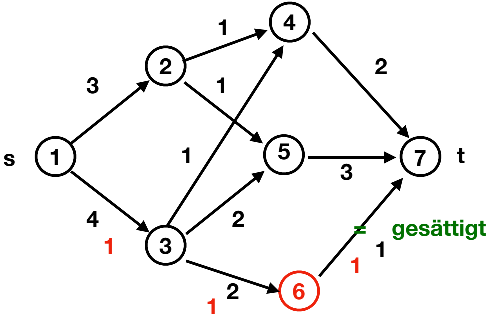


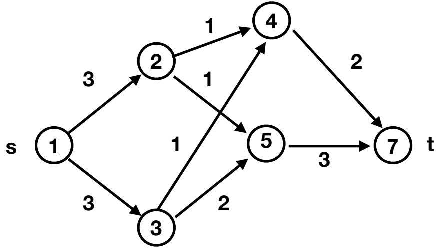
Tabelle der Potenziale:

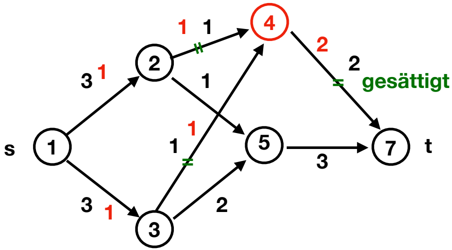
j	1	2	3	4	5	6	7
P_{in}	∞	3	4	2	3	2	6
P_{out}	7	2	5	2	3	1	∞
P	7	2	4	2	3	1	6

(2.1) Knoten mit kleinstem Potential: $j = 6$

(2.2) Fluss $s \to j$ mit Stärke $P(j) = 1$
Fluss $j \to t$ mit Stärke $P(j) = 1$.

Algorithmus von Dinic: Details zu Schritt (2) – Beispiel





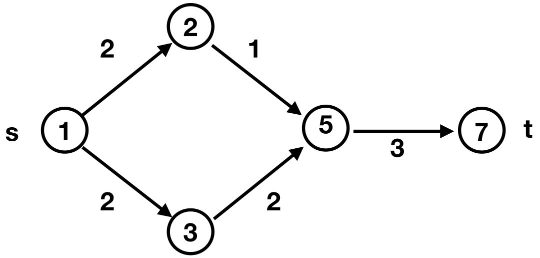
2. Iteration: Update der Potenziale:

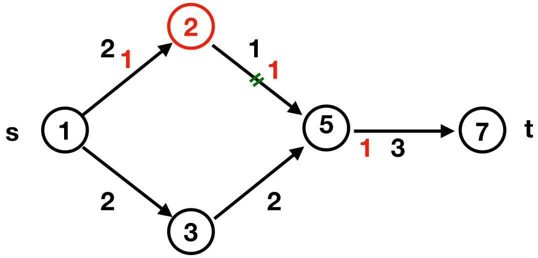
j	1	2	3	4	5	7
P_{in}	∞	3	3	2	3	5
P_{out}	6	2	3	2	3	∞
P	6	2	3	2	3	5

(2.1) Knoten mit kleinstem Potential: $j = 4$

(2.2) Fluss $s \to j$ mit Stärke $P(j) = 2$
Fluss $j \to t$ mit Stärke $P(j) = 2$.

Algorithmus von Dinic: Details zu Schritt (2) – Beispiel





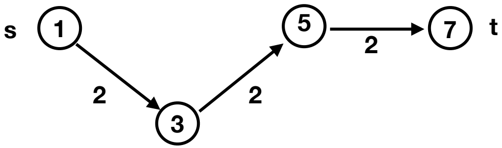
3. Iteration: Update der Potenziale:

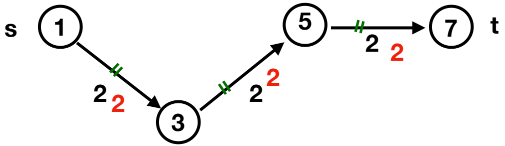
j	1	2	3	5	7
P_{in}	∞	2	2	3	3
P_{out}	4	1	4	3	∞
P	4	1	2	3	3

(2.1) Knoten mit kleinstem Potential: $j = 2$

(2.2) Fluss $s \to j$ mit Stärke $P(j) = 1$
Fluss $j \to t$ mit Stärke $P(j) = 1$.

Algorithmus von Dinic: Details zu Schritt (2) – Beispiel





4. Iteration: Update der Potenziale:

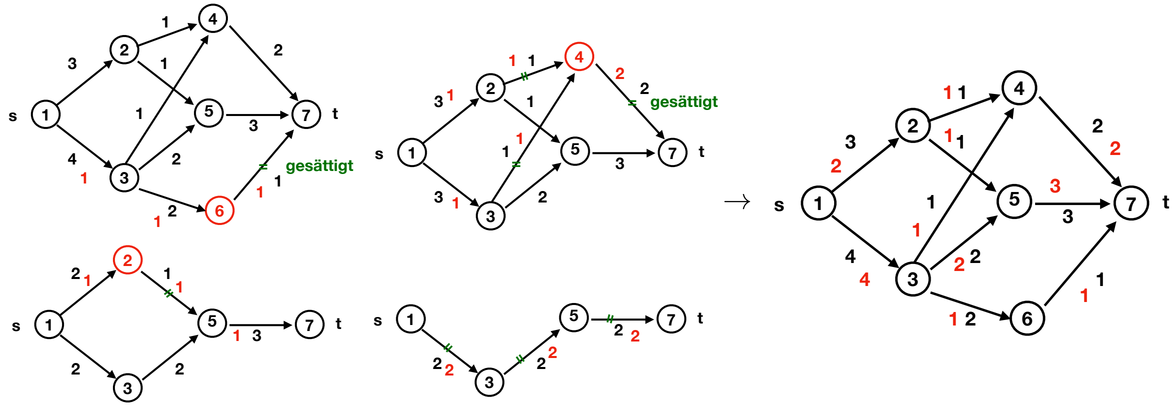
j	1	3	5	7
P_{in}	∞	2	2	2
P_{out}	2	2	2	∞
P	2	2	2	2

(2.1) Wähle Knoten mit kleinstem Potential: $j = 1$

(2.2) Fluss $j = 1 \rightarrow s$ mit Stärke $P(j) = 2$
Fluss $j = 1 \rightarrow t$ mit Stärke $P(j) = 2$.

Kanten gesättigt $\rightsquigarrow$ löschen $\rightsquigarrow$ Ende.

Algorithmus von Dinic: Details zu Schritt (2) – Beispiel: Gesamtfluss



Gesamtfluss (rechts) sättigt in jedem (s-t)-Weg eine Kante.

Inhalt

Algorithmus von Dinic: Aufwand

Lemma

Die Länge des (s-t)-Weges im Schichtgrafen (d. h. die Anzahl der Schichten) wächst in jeder Phase (außer der letzten) mindestens um eine. Also gibt es maximal $|V|$ Phasen.

Beweis-Idee:

- *In jedem Weg in (V, E_f) wird eine Kante gesättigt.*
- *Diese Kante fehlt im nächsten Schritt als nützliche Kante.*
- *Es können aber nützliche Rückwärtskanten hinzukommen.*
- *Bezüglich der Schichten laufen diese im Graphen der nützlichen Kanten in Rückwärtsrichtung.*

Algorithmus von Dinic: Aufwand

Satz

Der Algorithmus von Dinic hat eine Laufzeit von $\mathcal{O}(|V|^2|E|)$ (und damit effizienter als der Algorithmus von Edmonds-Karp mit $\mathcal{O}(|E|^2|V|)$).

Beweis-Idee:

- *Schritt (2.1), Potenziale bestimmen: $\mathcal{O}(|E|)$.*
- *Schritt (2.2): In jeder Iteration fällt ein Knoten weg,*
- *Senden des Flusses: maximal alle Knoten benutzt. $\rightsquigarrow$ Schritt (2) insgesamt: $\mathcal{O}(|V||E|)$*
- *Schichtgraf wird in Schritt (1) maximal $|V|$-mal aufgebaut. $\rightsquigarrow$ $\mathcal{O}(|V|^2|E|)$.*

Bemerkung

Für dicht besetzte Graphen gilt damit für den Aufwand: Edmonds-Karp $\leftrightarrow$ Dinic $\mathcal{O}(|V|^5) \leftrightarrow \mathcal{O}(|V|^4)$, für dünn besetzte immmer noch $\mathcal{O}(|V|^3) \leftrightarrow \mathcal{O}(|V|^3)$.

Lernziele

- Den Algorithmus von Dinic erklären ...
- ... und am Beispiel durchführen können.
- Dazu Details von Schritt (2) – Konstruktion eines zulässigen Flusses, der in jedem (s-t)-Weg eine Kante sättigt.
- Die Aussage über die Komplexität des Algorithmus von Dinic und die Beweisidee dazu kennen ...
- ... und sie im Vergleich zu den bisherigen Algorithmen einordnen können.