Push/Relabel Methode (Goldberg-Tarjan)

ANNAHME: $D = (V, A)$ ist symmetrisch, d.h.

$$(u, v) \in A \qquad \Longleftrightarrow \qquad (v, u) \in A.$$

[Bem.: Kann man durch Hinzufügen von Kanten (x, y) mit $c(x, y) = 0$ immer erreichen.]

Pseudofluß: $f : A \to \mathbb{R}$ mit

$$\begin{aligned} f(u,v) &\leq c(u,v) \quad \text{für alle}(u,v) \in A \quad \text{und} \\ f(u,v) &= -f(v,u) \quad \text{für alle}(u,v) \in A. \end{aligned}$$

Überschuß (engl. *excess*) in einem Knoten $v \in V$:

$$e_f(v) := \sum_{u:(u,v)\in A} f(u,v)$$

Präfluß (engl. *preflow*): Pseudofluß mit

$$e_f(v) \geq 0 \qquad \text{für alle } v \in V \setminus \{s, t\}.$$

aktiver Knoten: Knoten $v \in V \setminus \{s, t\}$ mit $e_f(v) > 0$

Restkapazität: $c_f(a) = c(a) - f(a) \qquad$ für alle $a \in A$.

Restgraph: $D_f = (V, A_f)$ wobei
$A_f = \{a \in A \mid c_f(a) > 0\}$.

(zulässige) Markierung bzgl. f: Fkt. $d : V \to \mathbb{N}_0$ mit

$$d(s) = n,\ d(t) = 0 \text{ und}$$

$$d(u) \leq d(v) + 1 \qquad \text{für alle } (u, v) \in A_f.$$

Basisalgorithmus:

// Initialisierung
for all $(u, v) \in A$ **do**
    $f[u, v] = 0$;
    if $u = s$ **then** $f[u, v] = c[u, v]$;
    if $v = s$ **then** $f[u, v] = -c[v, u]$;
for all $v \in V$ **do**
    $e_f[v] = \sum_{w\,:\,(w,v)\in A} f[w, v]$;
    if $v = s$ **then** $d[v] = n$ **else** $d[v] = 0$;
// Schleife
while $\exists$ aktiver Knoten **do**
    Führe eine (zulässige) Update-Operation durch;

Update-Prozeduren

$\textsc{Push}(v, w)$

Anwendbarkeit: v aktiv und $d[v] = d[w] + 1$;
Aktion: Sende $\delta = \min\{e_f(v), c_f(v, w)\}$ Flußeinheiten
von v nach w;

$\textsc{Relabel}(v)$

Anwendbarkeit: v aktiv und
$((v, w) \notin A_f$ oder $d[w] \geq d[v]) \quad \forall w \in \Gamma^+(v)$;
Aktion: $d[v] = \min\{d[w] + 1 \mid w \in \Gamma^+(v)$ mit $(v, w) \in A_f\}$;

$\textsc{Push}/\textsc{Relabel}(v)$

Anwendbarkeit: v aktiv

Sei $\{v, w\}$ die derzeitige *current edge* von v;

if $\textsc{Push}(v)$ anwendbar **then**

  $\textsc{Push}(v)$

else

  if $\{v, w\}$ letzte Kante der Kantenliste von v **then**

    Mache erste Kante der Kantenliste zur *current edge*;

    $\textsc{Relabel}(v)$;

  else

    Ersetze *current edge* durch die nächste Kante der Kantenliste;

Strategien

- FIFO – aktive Knoten werden in einer Queue gehalten, nach dem *first-in first-out* Prinzip
  Laufzeit: $\mathcal{O}(n^3)$

- HIGHEST LABEL – es wird jeweils der aktive Knoten mit maximalem d-Wert betrachtet
  Laufzeit: $\mathcal{O}(n^2\sqrt{m})$

- EXCESS SCALING – es wird nur Fluß von Knoten mit „genügend“ großem Überschuß zu Knoten mit “genügend“ kleinem Überschuß verschickt
  Laufzeit: $\mathcal{O}(nm + n^2 \log c_{max})$

<u>Kor 1.12:</u> Basisalg. terminiert mit einem maximalen Fluß

<u>Lem 1.13:</u> # relabelings $\leq 2n^2$

<u>Lem 1.14:</u> # sat. pushs $\leq nm$

<u>Lem 1.15:</u> # nicht sat. pushs $\leq 2n^2m$

<u>Satz 1.17:</u> Basisalg. mit $\textsc{Push}/\textsc{Relabel}$ hat Laufzeit $\mathcal{O}(n^2m)$

<u>Satz 1.18:</u> Basisalg. mit $\textsc{Push}/\textsc{Relabel}$ und FIFO-Strategie hat Laufzeit $\mathcal{O}(n^3)$

Basisalgorithmus mit FIFO-Strategie:

// Initialisierung

$\langle$ ... wie bisher ... $\rangle$

Füge zusätzlich alle aktiven Knoten in eine Queue Q ein;

// Schleife

while Q nicht leer **do**

    Entferne vordersten Knoten v von Q;

    repeat

        $\textsc{Push}/\textsc{Relabel}(v)$;

        Falls dadurch ein Knoten w aktiv wird, füge ihn am Ende von Q an;

    until $e_f(v) = 0$ oder $d(v)$ zugenommen hat;

    Falls v noch immer aktiv ist, füge v am Ende von Q an;