**Beweis** (2. Teil)

Damit durch eine Augmentierung ein Kreis im Residualnetzwerk entsteht, muß der augmentierende Pfad durch eine Kante des Kreises rückwärts gehen.

Sei $p$ ein augmentierender Pfad, durch den ein Kreis $K$ mit negativen Kosten entsteht. Die Kante $(u, v)$ liege auf $p$ und die Kante $(v, u)$ liege auf $K$.

Sei $p'$ folgender Pfad: Von $s$ entlang $p$ bis $u$, dann entlang $K$ bis $v$, dann entlang $p$ bis $t$.

Die Kosten von $p'$ sind dann kleiner als von $p$. Also kann $p$ kein augmentierender Pfad mit minimalen Kosten sein.

## Ein einfacher Algorithmus

Gesucht: Fluß mit Wert $B$ und minimalen Kosten

Initialisiere Fluß $f$ zu 0
**while** es gibt einen augmentierenden Pfad $p$ **do**
    finde einen augmentierenden Pfad $p$
        mit minimalen Kosten
    augmentiere $f$ entlang $p$ mit $\min\{B - |f|, c_f(p)\}$
**return** $f$

Durch das Minimum Cost Flow Theorem ist garantiert, daß der gefundene Fluß minimale Kosten hat.

## Die Edmonds–Karp–Variante

**for** each edge $(u, v) \in E$ **do**
  $f(u, v) := 0$
  $f(v, u) := 0$
**while** there exists a path from $s$ to $t$ in $G_f$ **do**
  $p :=$ a shortest, minimum cost path from $s$ to $t$ in $G_f$
  $c_f(p) := \min\{\, c_f(u, v) \mid (u, v) \text{ is in } p \,\}$
  **for** each edge $(u, v)$ in $p$ **do**
    $f(u, v) := f(u, v) + \min\{B - |f|, c_f(p)\}$
    $f(v, u) := -f(u, v)$
**return** f

**Lemma D'**

Gegeben ist ein $s$-$t$-Netzwerk $G = (V, E)$. Sei $f$ ein Fluß, der während der Ausführung der Edmonds–Karp–Variante vorkommt und $f'$ der Fluß nach der anschließenden Augmentierung.

Dann gilt: $\delta'(s, v) \geq \delta(s, v)$ für alle $v \in V$.

Hierbei sind $\delta(u, v)$ (bzw. $\delta'(u, v)$) die Kosten des billigsten Pfads von $u$ nach $v$ in $G_f$ (bzw. in $G_{f'}$).

**Beweis:**

Wie Lemma D

**Theorem**

Die Edmonds–Karp–Variante findet einen Fluß mit Wert $B$ und minimalen Kosten in endlich vielen Schritten.

**Beweis**

- Solange die Kosten der augmentierenden Pfade gleichbleiben, werden unter ihnen kürzeste gewählt und der Algorithmus arbeitet wie der Edmonds–Karp–Algorithmus.

  $\Rightarrow$ Nach $O(|E| \cdot |V|)$ Iterationen, wird ein Pfad mit höheren Kosten gewählt (Lemma D').

- Es gibt nur endlich viele, mögliche augmentierende Pfade und damit nur endlich viele mögliche Kosten.

## Bipartites Matching maximalen Gewichts

**Gegeben:**

Ein bipartiter, ungerichteter Graph $(V_1, V_2, E)$ und eine Gewichtsfunktion $b\colon E \to \mathbf{R}_0^+$.

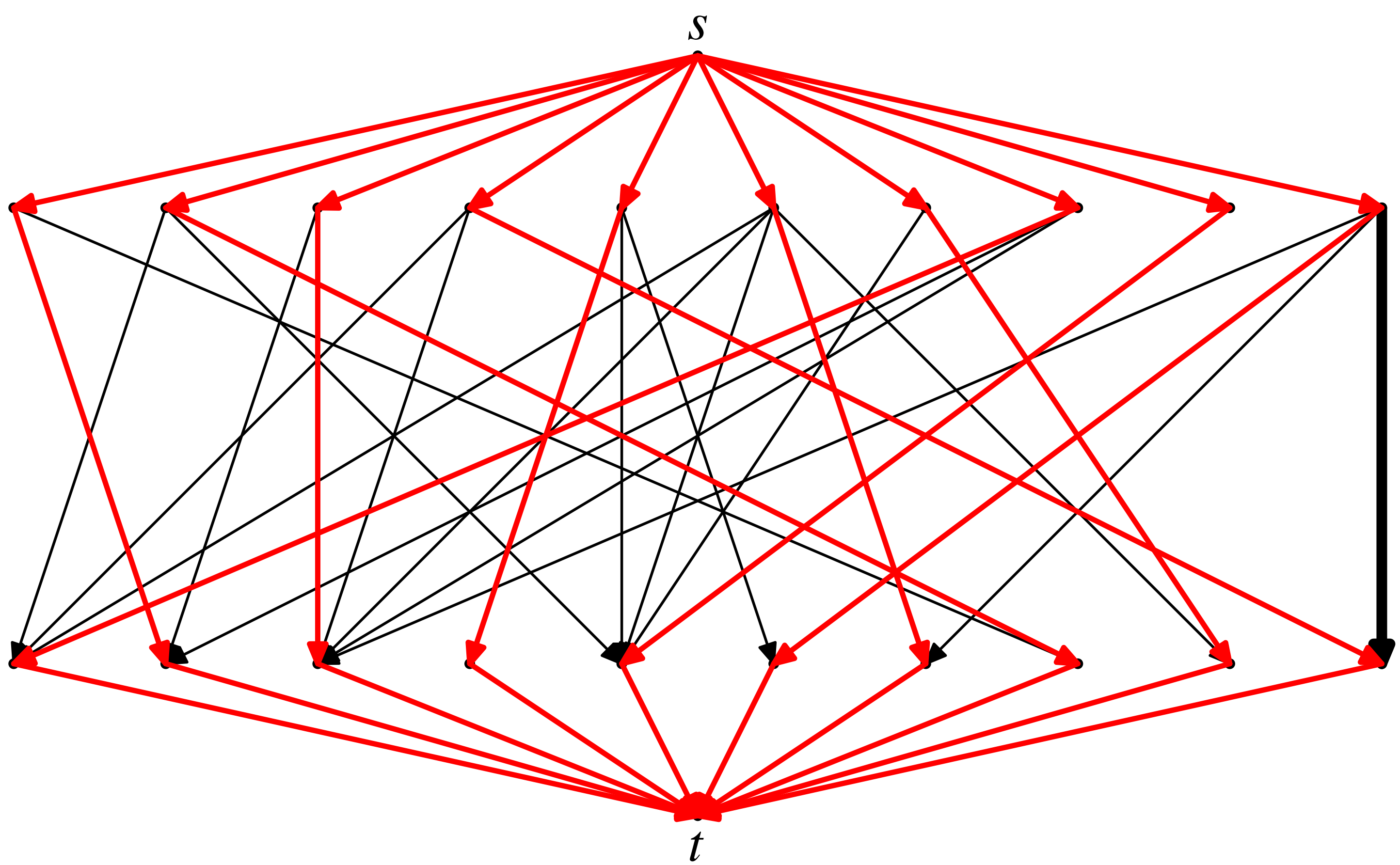
**Gesucht:**

Ein Matching mit maximalem Gewicht.

Das Gewicht eines Matchings $M$ ist

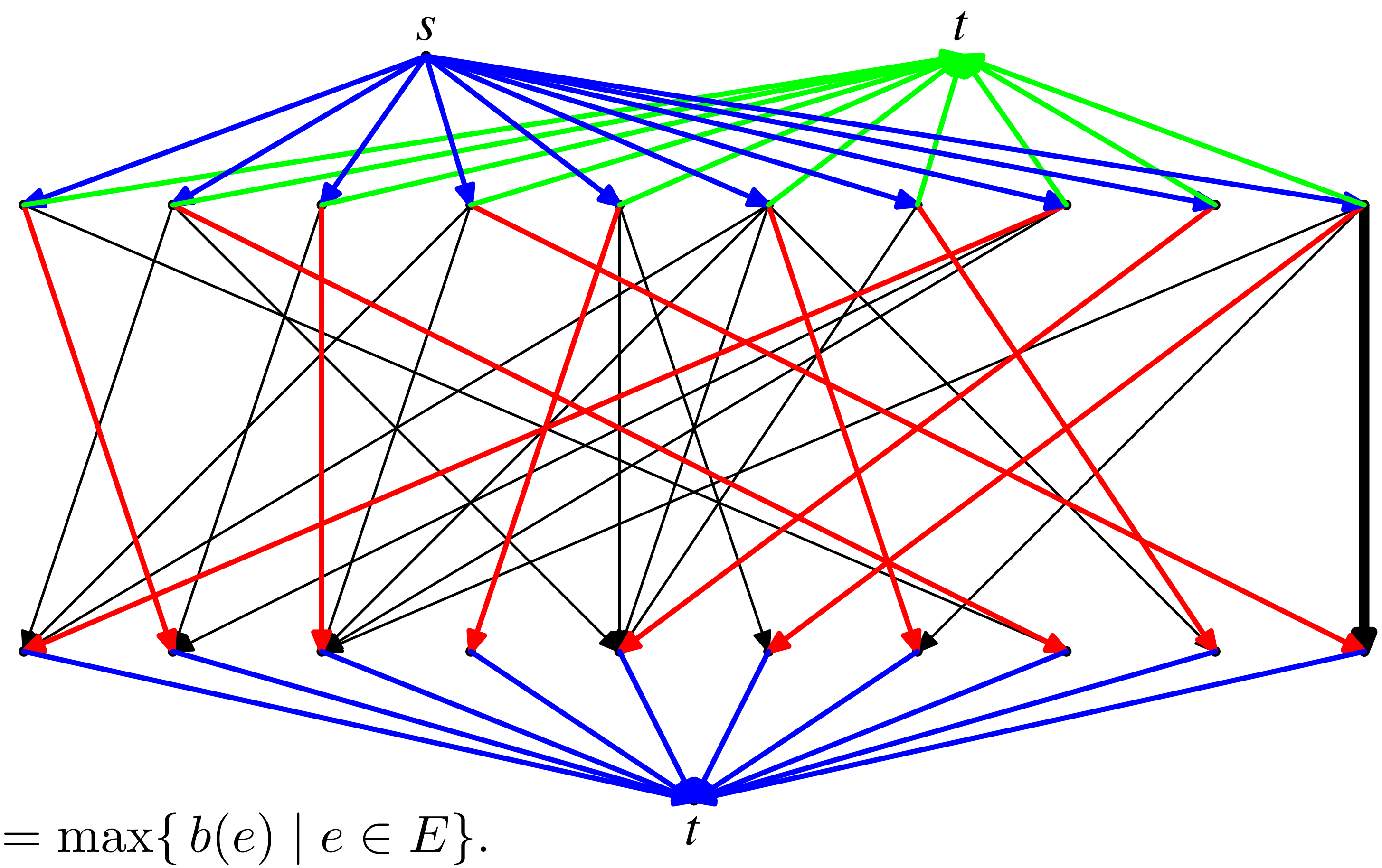
$$b(M) = \sum_{e \in M} b(e).$$

## Beispiel




Alle Kanten haben Gewicht 1, nur die dicke Kante hat 3.

Hat das perfekte Matching auch maximales Gewicht?

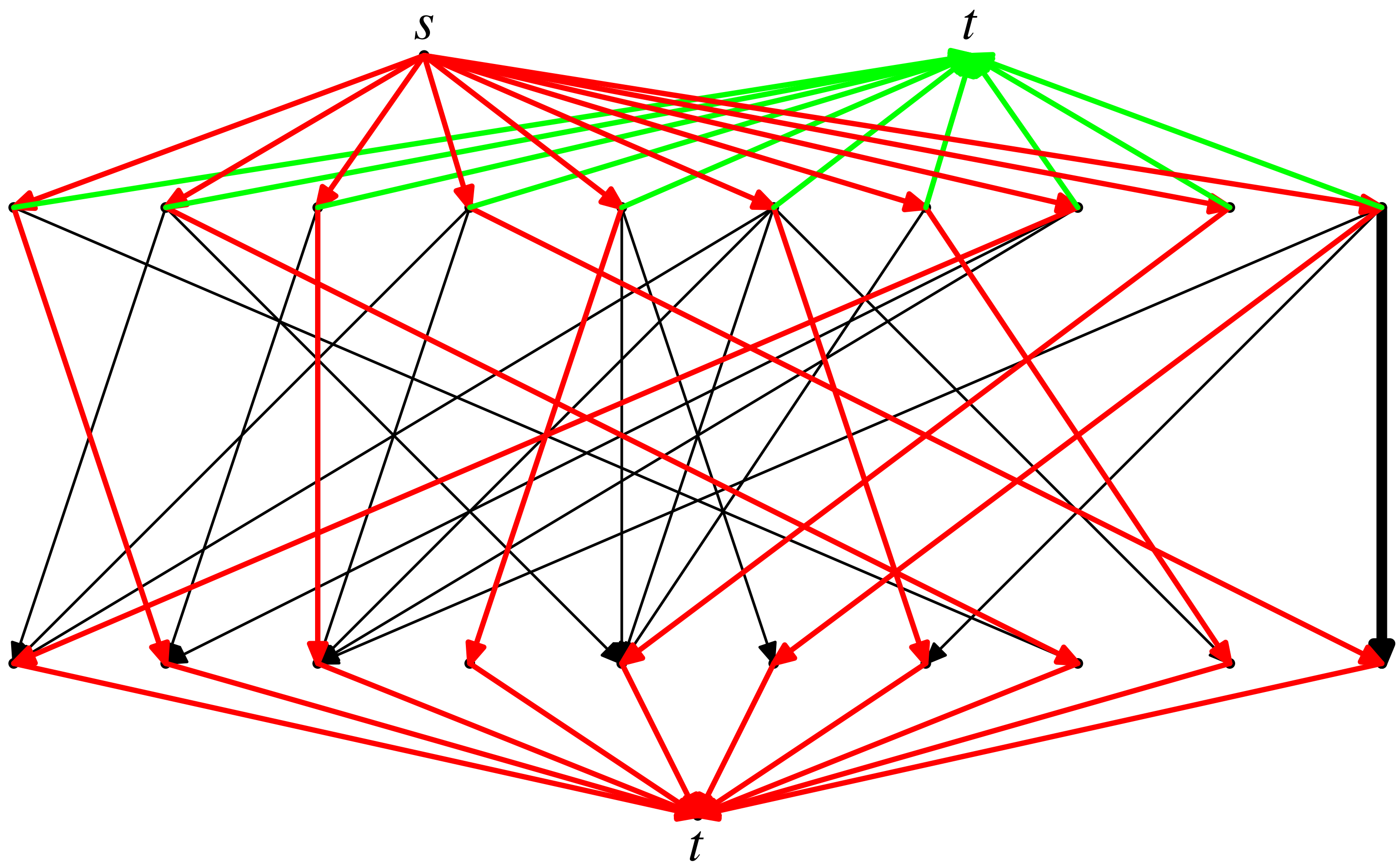



$w^* = \max\{\, b(e) \mid e \in E \}$.

Blaue Kanten haben Kosten 0, grüne Kanten haben Kosten $w^*$, eine Matching-Kante $e$ hat Kosten $w^* - b(e)$.

Matching mit Gewicht $b \Longrightarrow$ Fluß mit Kosten $nw^* - b$.

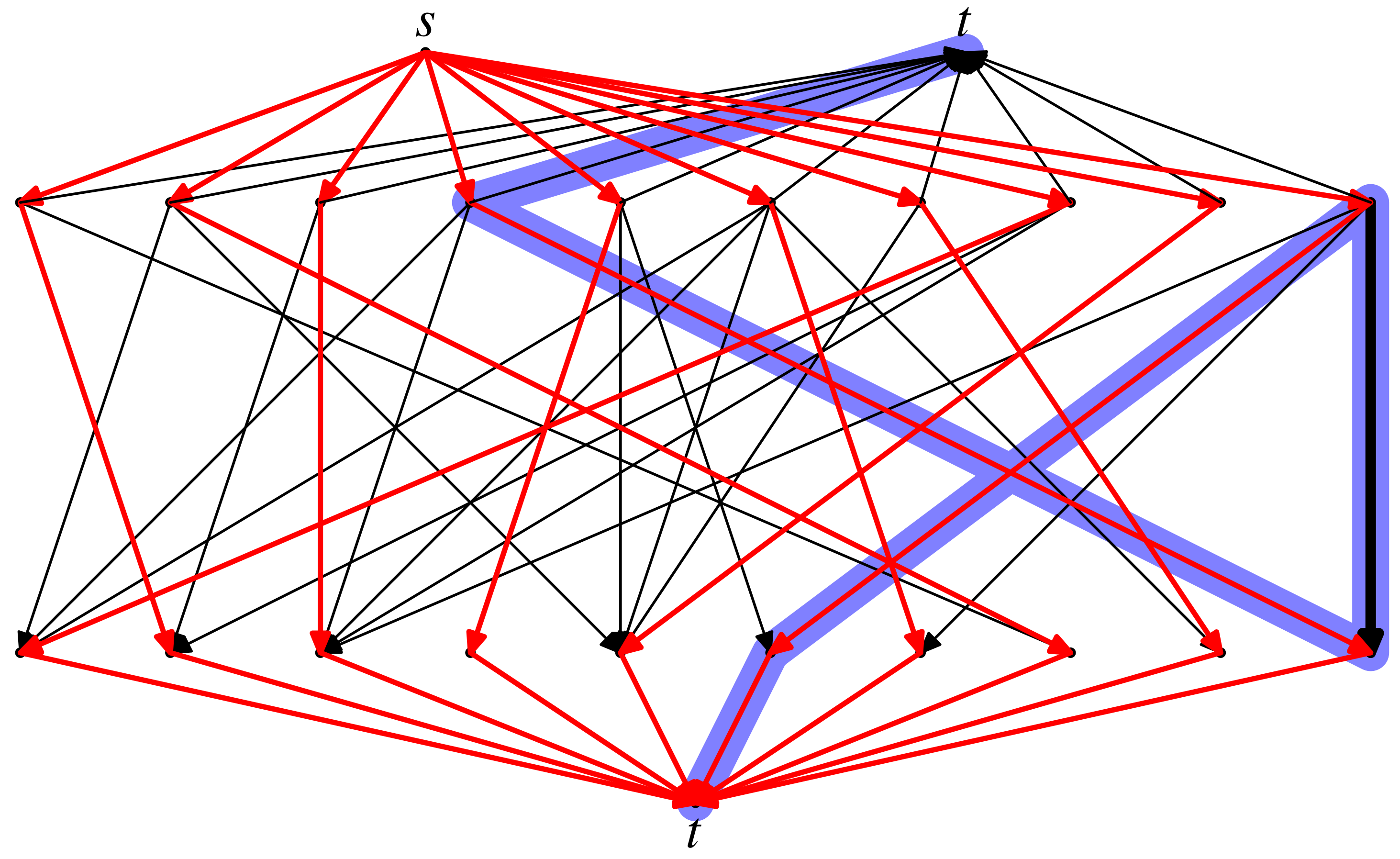
## Beispiel




Dem perfekten Matching entspricht der rote Fluß mit Wert 10.

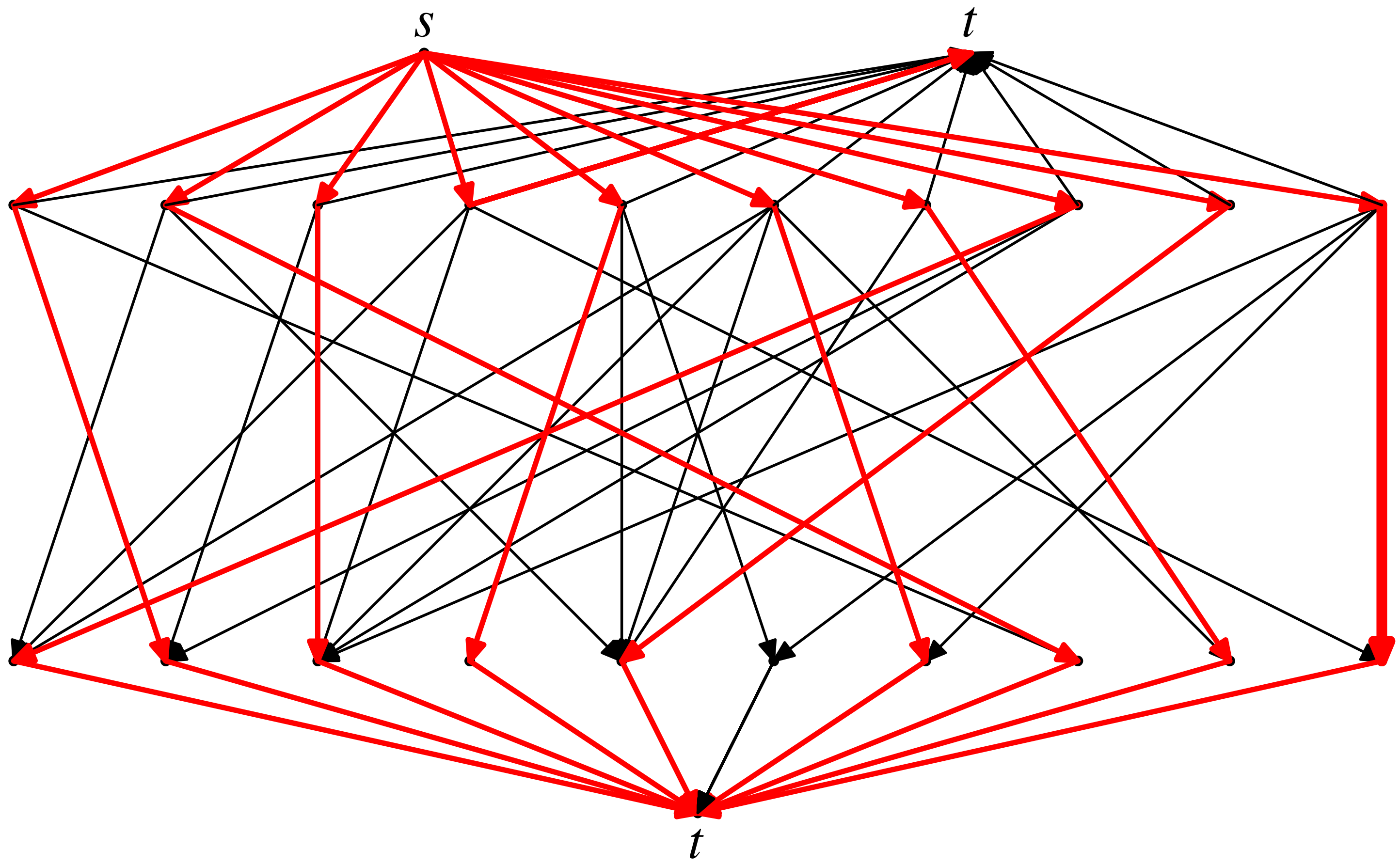
Wie groß sind die Kosten? ($w^* = 3$)

## Beispiel




Die Kosten sind nicht minimal, denn es gibt einen Kreis in $G_f$ mit negativen Kosten.

## Beispiel




Jetzt hat der Fluß Wert 10 und minimale Kosten. Das entsprechende Matching ist kleiner, hat aber maximales Gewicht.

## Bipartites Matching maximalen Gewichts

Bipartites Matching maximalen Gewichts läßt sich mittels Flusses minimaler Kosten lösen:

Die Kosten des Flusses sind

- $(n - |M|)w^*$ und
- $\sum_{e \in M} (w^* - b(e))$

Zusammen sind es $nw^* - b(M)$ (mit $n = |V_1| \geq |V_2|$)

Dieser Wert ist minimal, wenn $b(M)$ maximal ist.$\square$