



Übungsblatt 9.

Bearbeiten bis: Montag, 19.05.2025

Aufgabe 1 (Flussnetzwerke).

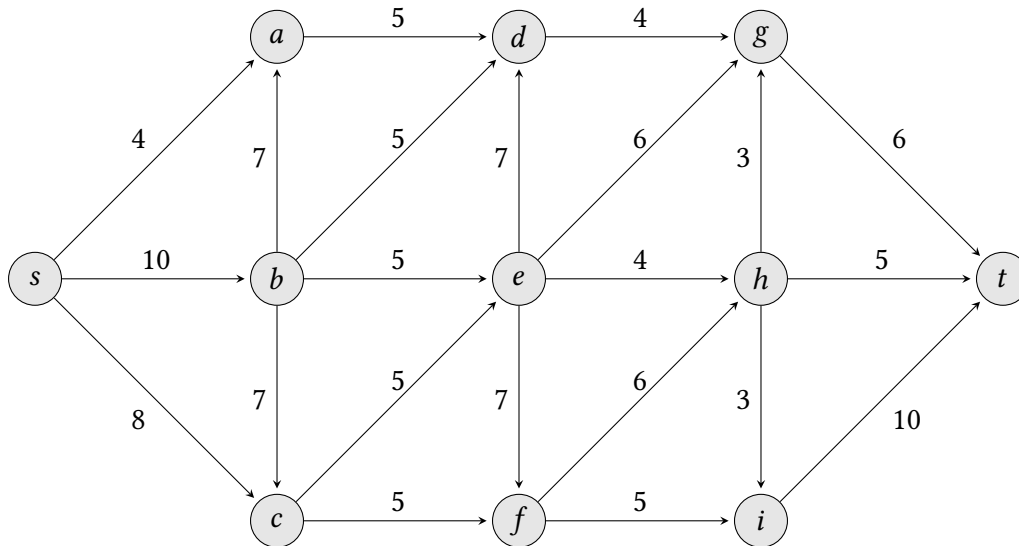


Abbildung 1: Netzwerk N_1

- (a) Berechnen Sie für das Flussnetzwerk N_1 in Abbildung 1 einen Maximalfluss f und begründen Sie ihn. Die angegebenen Zahlen geben die Kapazität der jeweiligen Kante an. Zu jedem Schritt sollen die augmentierten Wege angegeben werden.
- (b) Geben sie einen Schnitt minimaler Kapazität an.

Aufgabe 2 (Ford-Fulkerson | 4 Punkte). Gegeben sei das Netzwerk N_2 in Abbildung 2 mit Fluss f .

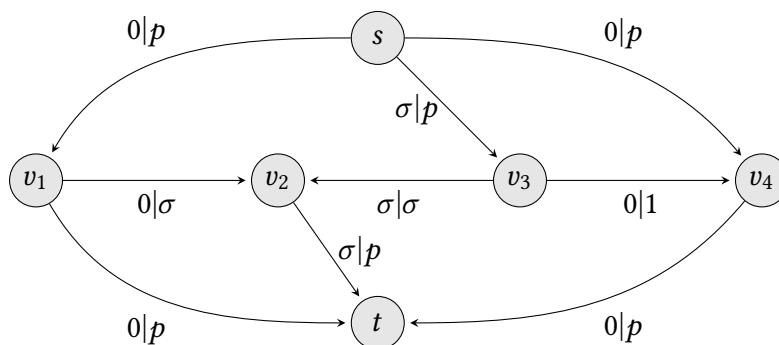


Abbildung 2: Netzwerk N_2

Dabei seien $\sigma = \frac{\sqrt{5}-1}{2}$ und $p = \frac{1}{1-\sigma}$. Der gegebene Fluss resultiert als Beispiel aus dem Nullfluss durch Augmentierung entlang des Wegs $\pi = s, v_3, v_2, t$. Zeigen Sie, dass es eine Folge

von augmentierten Wegen $\pi_1, \pi_2, \pi_1, \pi_3, \pi_1, \pi_2, \pi_1, \pi_3, \dots$ gibt, so dass der Ford-Fulkerson-Algorithmus nicht terminiert. Dazu beachte man, dass $\sigma^n = \sigma^{n+1} + \sigma^{n+2}$ gilt.

Hinweis: $p = \sum_{k=0}^{\infty} \sigma^k = \frac{1}{1-\sigma}$ ist der Wert der geometrischen Reihe von σ .

Aufgabe 3 (Edmonds-Karp I | 4 Punkte). Zeigen Sie graphisch, analog zum Beispiel in der Vorlesung und im Skript, die Schritte des Edmonds-Karp-Algorithmus auf dem Netzwerk N_3 in Abbildung 3. Zeichnen Sie zu jeder Iteration den Digraphen mit dem zugehörigen Fluss und geben Sie den kürzesten augmentierenden Pfad an.

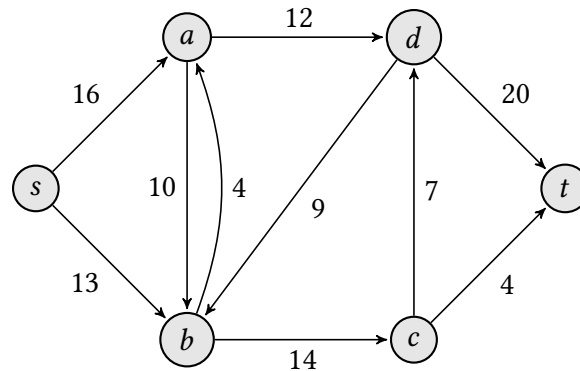


Abbildung 3: Netzwerk N_3

Aufgabe 4 (Edmonds-Karp II). Implementieren Sie den Edmonds-Karp-Algorithmus. Nutzen Sie hierfür die modifizierte Breitensuche aus Algorithmus `search_ek` um den kürzesten augmentierenden $s-t$ Pfad zu finden. Testen Sie Ihre Implementierung am Netzwerk N_4 in Abbildung 4. Geben Sie für jede Iteration der `while`-Schleife im Algorithmus `edmonds_karp` den berechneten kürzesten augmentierenden Pfad aus.

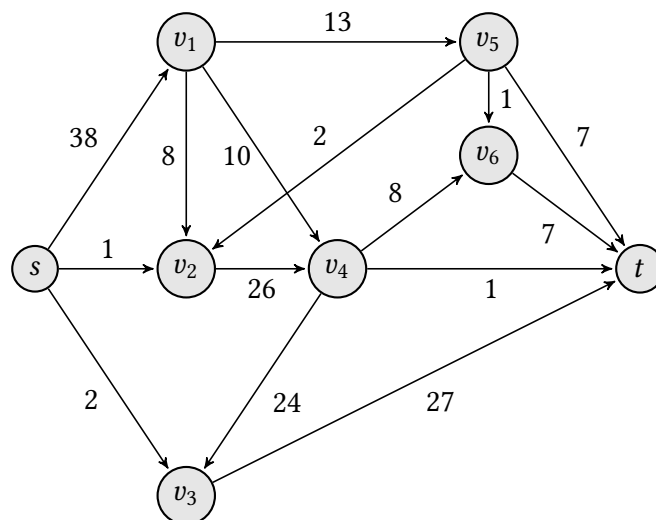


Abbildung 4: Netzwerk N_4

Algorithmus edmonds_karp(A, C)

Input: Matrix der Nachbarschaftsknoten A , wobei s der erste und t der letzte Knoten ist, Matrix der Kapazitäten C , welche dieselbe Struktur wie A besitzt

Output: maximale Flussfunktion F , welche dieselbe Struktur wie A besitzt

```
 $F = 0$   
[ $p, d$ ] = search_ek( $A, C, F$ )  
while  $d < \infty$  do  
    Augmentiere  $p$  um  $d$   
    [ $p, d$ ] = search_ek( $A, C, F$ )  
end while
```

Algorithmus search_ek(A, C, F)

Input: Matrix der Nachbarschaftsknoten A , wobei s der erste und t der letzte Knoten ist, Matrix der Kapazitäten C , welche dieselbe Struktur wie A besitzt, aktuelle Flussfunktion F

Output: $p(v)$ Vorgänger von v , $d \in \mathbb{R}$ wobei: falls ein kürzester augmentierter s - t Pfad existiert, ist d das Minimum über $C(e) - F(e)$, falls e Vorwärtskante, und $d = F(e)$, falls e Rückwärtskante; falls kein kürzester augmentierter s - t Pfad existiert, ist $d = \infty$.

```
 $r = s$   
 $q = s$   
 $b(v) = \infty, \forall v \in V$   
 $d = \infty$   
while  $q \neq []$  do  
     $v = q[1]$   
    for all  $w = A[i][v] \notin r$  mit  $C[i][v] - F[i][v] > 0$  do  
         $p(w) = v$   
         $b(w) = \min\{C[i][v] - F[i][v], b(v)\}$   
        if  $w = t$  then  
             $d = b(w)$   
            return [ $p, d$ ]  
        end if  
         $r = [r, w]$   
         $q = [q, w]$   
    end for  
    for all  $w \notin r$  mit  $v = A[i][w]$  und  $F[i][w] > 0$  do  
         $p(w) = v$   
         $b(w) = \min\{F[i][w], b(v)\}$   
        if  $w = t$  then  
             $d = b(w)$   
            return [ $p, d$ ]  
        end if  
         $r = [r, w]$   
         $q = [q, w]$   
    end for  
     $q = q[2 : \text{end}]$   
end while
```
