

Graphalgorithmen

Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

SS 2021

Matchings

Definition. Sei $G = (V, E)$ ein Graph und sei $M \subseteq E$.

- M heißt **Matching** in G , falls je zwei Kanten $e \neq e' \in M$ **unabhängig** sind, d.h. $e \cap e' = \emptyset$
- Die **Matchingzahl** von G ist

$$\mu(G) = \max\{|M| : M \text{ ist ein Matching in } G\}$$

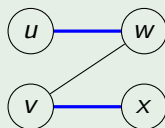
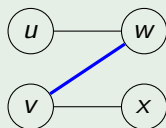
- Ein Knoten $v \in V$ heißt **M -gebunden**, falls v Endpunkt einer Kante $e \in M$ (also $v \in \cup M$) ist und sonst **M -frei**
- Wir sagen auch, **M bindet v** bzw. **M lässt v frei**
- Ein Matching M heißt **perfekt**, falls alle Knoten in G M -gebunden sind (also $V = \cup M$ ist)
- Ein Matching M heißt **maximal** (engl. *maximum*), falls $|M| = \mu(G)$ ist
- M heißt **gesättigt** (engl. *maximal*), falls es in keinem größeren Matching enthalten ist

Matchings

- Offenbar ist M genau dann ein Matching, wenn $|\cup M| = 2|M|$ ist
- Das Ziel besteht nun darin, ein maximales Matching M zu finden

Beispiel

- Ein gesättigtes Matching muss nicht maximal sein:



- $M = \{\{v, w\}\}$ ist gesättigt, da es sich nicht erweitern lässt
- M ist jedoch kein maximales Matching, da $M' = \{\{u, w\}, \{v, x\}\}$ ein größeres Matching ist
- Die Greedy-Methode, ausgehend von $M = \emptyset$ solange Kanten zu M hinzuzufügen, bis sich M nicht mehr zu einem größeren Matching erweitern lässt, funktioniert also nicht

Matchings

Satz

In einem bipartiten Graphen $G = (U, W, E)$ lässt sich ein maximales Matching in Zeit $O(m\sqrt{n})$ bestimmen

Beweis.

- Wir konstruieren zu G das Netzwerk $N = (V, E', s, t, c)$ mit der Knotenmenge $V = U \cup W \cup \{s, t\}$ und der Kantenmenge

$$E' = \{(u, w) \in U \times W \mid \{u, w\} \in E\} \cup \{(s, u), (w, t) \mid u \in U, w \in W\},$$
 wobei $c(e) = 1$ für alle $e \in E'$ gilt
- Da alle Knoten $u \in U \cup W$ den Durchsatz $D(u) \leq 1$ in N haben, liefert jeder Fluss f in N ein Matching $M = \{\{u, w\} \in E \mid f(u, w) = 1\}$ in G mit $|M| = |f|$ und umgekehrt
- Mit Dinitz lässt sich in N unter Verwendung von `blockfluss1` ein maximaler Fluss (und damit ein maximales Matching) in Zeit $O(m\sqrt{n})$ bestimmen, da der Durchsatz aller Knoten (außer s und t) 1 ist □

- In den Übungen werden wir sehen, wie sich die Laufzeit durch eine verbesserte Analyse sogar durch $O(m\sqrt{\mu})$ begrenzen lässt
- Die Konstruktion im Beweis des vorigen Satzes lässt sich nicht ohne Weiteres auf Graphen verallgemeinern, die nicht bipartit sind
- Wir werden jedoch sehen, dass sich manche bei den Flussalgorithmen verwendete Ideen auch für Matchingalgorithmen einsetzen lassen
- So lassen sich Matchings, die nicht maximal sind, ähnlich vergrößern wie dies bei Flüssen durch einen Zunahmepfad möglich ist

Definition. Sei $G = (V, E)$ ein Graph und sei M ein Matching in G .

- Ein Pfad $P = (u_0, \dots, u_\ell)$ in G der Länge $\ell \geq 1$ heißt **M -alternierend**, falls für $i = 1, \dots, \ell - 1$ gilt:

$$e_i = \{u_{i-1}, u_i\} \in M \iff e_{i+1} = \{u_i, u_{i+1}\} \notin M$$

- Ein Kreis $C = (u_1, \dots, u_\ell, u_1)$ in G heißt **M -alternierend**, falls der Pfad $P = (u_1, \dots, u_\ell)$ M -alternierend ist und zudem gilt:

$$\{u_1, u_2\} \in M \iff \{u_1, u_\ell\} \notin M$$

- Ein M -alternierender Pfad $P = (u_0, \dots, u_\ell)$ heißt **M -vergrößernder Pfad** (oder einfach **M -Pfad**), falls beide Endpunkte von P M -frei sind

Matchings

Satz (Lemma von Berge)

Ein Matching M in G ist genau dann maximal, wenn es keinen M -Pfad in G gibt

Beweis.

- Ist $P = (u_0, \dots, u_l)$ ein M -Pfad, so liefert $M' = M \Delta P$ ein Matching der Größe $|M'| = |M| + 1$ in G , wobei wir P als Menge $\{\{u_{i-1}, u_i\} \mid i = 1, \dots, l\}$ seiner Kanten auffassen
- Ist dagegen M nicht maximal und M' ein größeres Matching, so betrachten wir die Kantenmenge $M \Delta M'$
- Da jeder Knoten in dem Graphen $G' = (V, M \Delta M')$ höchstens den Grad 2 hat, lässt sich G' in disjunkte Kreise und Pfade zerlegen
- Da diese Kreise und Pfade M -alternierend sind, und M' größer als M ist, muss mindestens einer dieser Pfade ein M -Pfad sein □

Es genügt also, für das aktuelle Matching M einen M -Pfad zu finden

Der Algorithmus von Edmonds

- Sei G ein Graph ohne isolierte Knoten und sei M ein Matching in G
- Die Prozedur `FindePfad` gibt einen M -Pfad in G zurück, falls das aktuelle Matching M nicht bereits maximal ist
- Da M nicht mehr als $n/2$ Kanten enthalten kann, muss diese Prozedur ausgehend von $M = \emptyset$ höchstens $\lfloor n/2 + 1 \rfloor$ -mal aufgerufen werden, um ein maximales Matching zu finden

Prozedur `FindePfad(V, E, M)`

```

1  Q := ∅
2  for all u ∈ V do
3    parent(u) := ⊥
4    if ∃ e ∈ M : u ∈ e then
5      zustand(u) := 2      // unerreicht
6    else
7      zustand(u) := 0      // gerade
8      root(u) := u
9      Q := Q ∪ {(u, v) | {u, v} ∈ E}

```


Prozedur $\text{FindePfad}(V, E, M)$ (Fortsetzung)

```
11 while  $Q \neq \emptyset$  do  
12   entferne eine Kante  $(u, v)$  aus  $Q$   
13   if  $\text{zustand}(v) = 2$  then  
14      $\text{parent}(v) := u$ ;  $\text{parent}(M(v)) := v$ ;  $\text{zustand}(v) := 1$  // ungerade  
15      $\text{zustand}(M(v)) := 0$ ;  $\text{root}(M(v)) := \text{root}(v) := \text{root}(u)$   
16      $Q := Q \cup \{(M(v), w) \mid \{M(v), w\} \in E \setminus M\}$   
17   if  $\text{zustand}(v) = 0$  then  
18     if  $\text{root}(u) = \text{root}(v)$  then // Blüte gefunden  
19       kontrahiere in  $W$  die Blüte  $C$  zu ihrer Basis  $b$  und speichere den  
20       Kreis  $C$  unter der Basis  $b$  ab  
21       füge zu  $Q$  für jede Kante  $\{c, a\} \in E$  mit  $c \in C$  ungerade und  
22        $a \notin C$  die Kante  $(b, a)$  hinzu  
23     else //  $M$ -Pfad gefunden  
24       setze die parent-Pfade  $P_u$  und  $P_v$  von  $u$  und  $v$  mit Hilfe  
25       der Kante  $\{u, v\}$  zu einem  $r_u$ - $r_v$ -Pfad  $P'$  zusammen  
26       und expandiere  $P'$  zu einem  $M$ -Pfad  $P$  in  $G$ ; return  $P$   
27 return  $\perp$ 
```

Der Algorithmus von Edmonds

- Die Prozedur `FindePfad` sucht wie folgt nach einem M -Pfad in G
- Jeder Knoten u hat einen von 3 Zuständen: gerade (0), ungerade (1) oder unerreicht (2)
- Zu Beginn sind alle M -freien Knoten gerade und alle M -gebundenen Knoten unerreicht
- Dann wird ausgehend von den M -freien Knoten als Wurzeln ein Suchwald W für G aufgebaut
- Hierzu wird Q als Menge aller Kanten (u, v) initialisiert, so dass u gerade (also eine Wurzel) und $\{u, v\} \in E$ ist
- In der `while`-Schleife werden dann die zu Q hinzugefügten Kanten $e = (u, v)$ besucht, wobei Kanten zu einem ungeraden Knoten v ignoriert werden

- Ist v unerreicht, so wird der aktuelle Suchwald W nicht nur um die Kante e , sondern auch um die Matching-Kante $\{v, M(v)\}$ erweitert, wobei $M(v)$ der Matchingpartner von v ist
- Zudem wechselt der Zustand von v von unerreicht zu ungerade und der von $M(v)$ von unerreicht zu gerade
- Somit erhält jeder erreichte Knoten v genau dann den Zustand gerade, wenn der Pfad zwischen v und seiner Wurzel r_v in W eine gerade Länge hat
- Um diesen Wurzelfad effizient berechnen zu können, wird die Funktion `parent` benutzt

Der Algorithmus von Edmonds

- Ist v dagegen wie u gerade, so gibt es zwei Unterfälle
- Haben u und v verschiedene Wurzeln $r_u \neq r_v$, so lassen sich die beiden parent-Pfade P_u von u und P_v von v mit Hilfe der Kante $\{u, v\}$ zu einem Pfad P' zusammensetzen, der die beiden M -freien Wurzeln r_u und r_v verbindet
- Da vor dem Auffinden von P' möglicherweise Blüten kontrahiert wurden (siehe unten), muss P' evtl. noch expandiert werden, um einen M -Pfad P in G zu erhalten

Der Algorithmus von Edmonds

- Im Fall $r_u = r_v$ befinden sich die beiden Knoten u und v im gleichen Suchbaum von W , d.h. die beiden parent-Pfade P_u und P_v enden beide in der Wurzel $r_u = r_v$
- Sei b der Knoten, in dem sich P_u und P_v trennen
- Da b (mindestens) 2 Kinder in W hat und ungerade Knoten nur ein Kind in W haben, muss b gerade sein
- Da auch u und v gerade sind, haben sie auf P_u bzw. P_v einen geraden Abstand zu b
- Der b - u -Teilpfad von P_u und der b - v -Teilpfad von P_v bilden also zusammen mit der Kante $\{u, v\}$ einen ungerichteten Kreis C ungerader Länge, der als **Blüte** mit der **Basis** b bezeichnet wird
- Da die ungeraden Knoten auf C durch die Kontraktion von C einen geraden Abstand zu ihrer Wurzel erhalten, wird für jede Kante $\{c, a\}$ in E mit $c \in C$ ungerade und $a \notin C$ die Kante (b, a) zu Q hinzugefügt

Der Algorithmus von Edmonds

- Zwar führt das Auffinden einer Blüte C nicht direkt zu einem M -Pfad
- Sie bedeutet dennoch einen Fortschritt, da sich G und W durch die Kontraktion von C zur Basis b verkleinern lassen
- In den kontrahierten Graphen G_C und W_C erbt b die Nachbarschaften aller Knoten in C zu den Knoten außerhalb von C
- Zudem wird zu Q für jede Kante $\{c, a\} \in E$ mit $c \in C$ ungerade und $a \notin C$ die Kante (b, a) hinzugefügt
- Entfernen wir aus M alle Kanten, die auf dem Kreis C liegen, so erhalten wir ein Matching M_C in G_C
- Das folgende Lemma zeigt, wie sich aus einem M_C -Pfad in G_C ein M -Pfad in G rekonstruieren lässt

Lemma. Sei C eine Blüte in G mit Basis b .

Dann ist jeder M_C -Pfad P_C in G_C zu einem M -Pfad P in G expandierbar

Beweis.

- Falls P_C nicht schon selbst ein M -Pfad in G ist, muss P_C eine Kante e enthalten, die in G fehlt
- Da durch die Kontraktion von C zu b in G_C nur solche Kanten neu entstehen, die die Basis b mit einem Knoten a außerhalb der Blüte C verbinden, muss e die Form $e = \{a, b\}$ haben
- Zudem muss a in G einen Nachbarn $c \neq b$ auf der Blüte C haben

Beweis (Fortsetzung).

- Von c aus führen auf dem Kreis C genau zwei Pfade zur Basis b , wovon nur einer den Knoten c über eine Matchingkante verlässt (also gerade Länge hat)
- Indem wir diesem Pfad die Kante $\{a, c\}$ hinzufügen, erhalten wir einen M -alternierenden a - b -Pfad P' in G
- Wir können also P_C zu einem M -Pfad P in G expandieren, indem wir die Kante $\{a, b\}$ durch den a - b -Pfad P' ersetzen □

Der Algorithmus von Edmonds

- Da sich die Anzahl der Knoten bei jeder Kontraktion einer Blüte mindestens um 2 verringert, können höchstens $n/2$ Blüten gefunden werden
- Bei Verwendung entsprechender Datenstrukturen zur Verwaltung der Blüten lässt sich die Prozedur `FindePfad` in Zeit $O(m)$ implementieren, was auf eine Gesamtlaufzeit von $O(nm)$ für den Algorithmus von Edmonds führt
- Tatsächlich lässt sich die Laufzeit noch auf $O(m\sqrt{\mu})$ verringern
- Dazu berechnet man ähnlich wie bei Verwendung von Diniz im bipartiten Fall pro Runde nicht nur einen M -Pfad, sondern in Zeit $O(m)$ eine maximale Menge knotendisjunkter M -Pfade, die alle eine minimale Länge haben
- Dann kann man wieder zeigen, dass $O(\sqrt{\mu})$ solcher Runden ausreichen, um ein maximales Matching zu finden
- Diese Strategie führt auf den Hopcroft-Karp-Algorithmus im bipartiten Fall und auf den Micali-Vazirani-Algorithmus für beliebige Graphen

Der Algorithmus von Edmonds

- Für den Beweis der Korrektheit des Edmonds-Algorithmus (genauer: zum Nachweis der Maximalität des berechneten Matchings) benötigen wir den Begriff der Odd Set Cover (OSC)
- Es ist leicht zu sehen, dass für jedes Matching M in einem Graphen G und jede Knotenüberdeckung C in G die Ungleichung $|M| \leq |C|$ gilt
- Wir werden sehen, dass es in jedem bipartiten Graphen sogar ein Matching M und eine Knotenüberdeckung C mit $|M| = |C|$ gibt
- Die Angabe einer Knotenüberdeckung C mit $|C| = |M|$ bietet also eine einfache Möglichkeit, die Maximalität von M nachzuweisen
- Dies geht jedoch nicht in allen Graphen, da z.B. der K_4 nur Matchings der Größe ≤ 2 und Knotenüberdeckungen der Größe ≥ 3 hat

Der Algorithmus von Edmonds

Definition Sei $G = (V, E)$ ein Graph.

Eine Menge $S = \{v_1, \dots, v_k, V_1, \dots, V_\ell\}$ von Knoten $v_1, \dots, v_k \in V$ und Teilmengen $V_1, \dots, V_\ell \subseteq V$ heißt **OSC** (engl. **odd set cover**) in G , falls

- es für jede Kante $e \in E$ einen Knoten $v_i \in S$ mit $v_i \in e$ oder eine Menge $V_j \in S$ mit $e \subseteq V_j$ gibt und
- alle Mengen $V_j \in S$ eine ungerade Größe $n_j = |V_j|$ haben.

Das **Gewicht** von S ist $w(S) = k + \sum_{j=1}^{\ell} (n_j - 1)/2$

Im Fall $\ell = 0$ ist $S = \{v_1, \dots, v_k\}$ also eine Knotenüberdeckung in G (oder kurz VC für engl. vertex cover)

Beispiel

K_4 hat auf der Knotenmenge $V = \{1, \dots, 4\}$ eine OSC $S = \{1, \{2, 3, 4\}\}$ vom Gewicht $w(S) = 1 + (3 - 1)/2 = 2 = \mu(K_4)$

Lemma

Für jedes Matching M in einem Graphen $G = (V, E)$ und jede OSC $S = \{v_1, \dots, v_k, V_1, \dots, V_\ell\}$ in G gilt $|M| \leq w(S)$

Beweis.

M kann für jeden Knoten $v_i \in S$ höchstens eine Kante e mit $v_i \in e$ und für jede Menge $V_j \in S$ höchstens $(n_j - 1)/2$ Kanten $e \subseteq V_j$ enthalten \square

Der Algorithmus von Edmonds

Satz

Falls M ein Matching für G ist und $\text{FindePfad}(G, M)$ keinen M -Pfad findet (also \perp zurückgibt), dann ist M maximal

Beweis.

- Um eine OSC S für G mit $w(S) = |M|$ zu finden, analysieren wir die Struktur des bei der erfolglosen Suche nach einem M -Pfad generierten Suchwalds $W = (V_W, E_W)$
- Sei $V_0 \subseteq V_W$ die Menge der geraden, $V_1 = \{u_1, \dots, u_k\} \subseteq V_W$ die der ungeraden und V_2 sei die Menge der unerreichten Knoten in W
- Weiter seien $b_1, \dots, b_\ell \in V_0$ die Knoten in W , zu denen die gefundenen Blüten kontrahiert wurden, und für $j = 1, \dots, \ell$ sei $C_j \subseteq V$ die Menge aller Knoten, die zu b_j kontrahiert wurden (d.h. von den Knoten in C_j ist nur noch b_j in V_W vorhanden)
- Zudem sei $V'_0 = V_0 \cup C_1 \cup \dots \cup C_\ell = V \setminus (V_1 \cup V_2)$ die Menge aller geraden und zu einem geraden Knoten kontrahierten Knoten in G