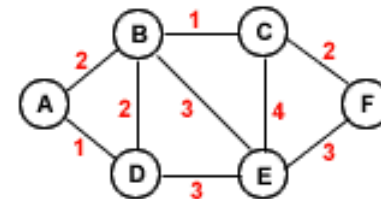




# Effiziente Algorithmen

Hartmut Klauck  
Universität Frankfurt  
SS 06  
22.6.





# Matchings

- Wir betrachten wieder das Matching Problem
- Ein augmentierender Pfad für ein Matching  $M$  auf einem bipartiten Graph ist ein Pfad, der in der Menge  $L$  bei einem Knoten beginnt, der zu keiner Kante in  $M$  inzident ist, eine Kante läuft, die nicht in  $M$  liegt, gefolgt von einer Kante in  $M$ , etc... und mit einer Kante nicht in  $M$  endet, bei einem Knoten in  $R$ , der zu keiner Kante in  $M$  inzident ist
- Offensichtlich können wir ein Matching wie folgt erweitern:
  - Bestimme einen augmentierenden Pfad
  - Wechsele die nicht-Matching-Kanten gegen die Matching-Kanten aus
- Höchstens  $n$  Iterationen.
- Laufzeit  $O(mn)$



# Matchings

- Ein besserer Ansatz besteht darin, eine maximale Menge von knotendisjunkten augmentierenden Pfaden zu bestimmen.
- Man kann zeigen: Es gibt nur  $n^{1/2}$  Iterationen in diesem Fall
- Hopcroft-Karp Algorithmus, Laufzeit somit  $O(n^{1/2}m)$



# Min Cuts

- Wir betrachten nun das Min-Cut Problem
- Gegeben sei ein ungewichteter ungerichteter Graph  $G$
- Wir suchen eine (nichttriviale) Partition  $L, R$  der Knotenmenge, bei der die Anzahl kreuzender Kanten minimal ist
- Unterschied zu  $s$ - $t$  Min-Cut: keine ausgezeichneten Knoten  $s, t$



# Ein randomisierter Ansatz

- Betrachte Multigraphen: es kann mehrere Kanten von  $u$  nach  $v$  geben
  - Min-Cuts sind analog definiert
1. Wähle eine Kante  $(u,v)$  zufällig, uniform aus allen Kanten
  2. Verschmelze Knoten  $u,v$ :
    - entferne alle Kanten von  $u$  nach  $v$
    - alle Kanten  $u$  nach  $w$  oder  $v$  nach  $w$  werden beibehalten
    - wir erhalten so einen neuen Multigraphen
  3. Stoppe wenn nur noch 2 Knoten übrig: diese definieren  $L,R$



# Ein randomisierter Ansatz

- Intuition:
  - Die Verschmelzungen reduzieren die Min-Cut Grösse nicht, erhöhen sie möglicherweise
  - $n-1$  Iterationen
  - Wenn eine kreuzende Kante eines Cuts kontrahiert wird, ist der Cut nicht mehr als Ausgabe möglich
  - Ein kleinerer Cut hat eine bessere Wahrscheinlichkeit, dass keine Kante des Cuts kontrahiert wird



# Ein randomisierter Ansatz

- Wir werden zeigen, dass ein beliebiger fester minimaler Schnitt das Ergebnis der Prozedur ist mit Wahrscheinlichkeit  $2/n^2$
- Daher können wir das ganze  $n^2$  mal wiederholen, und den kleinsten erhaltenen Schnitt ausgeben.
- Der Algorithmus kann so implementiert werden, dass die Laufzeit  $O(n^2 \log^2 n)$  ist
- Somit wird das Problem schneller gelöst als s-t- Min Cut



# Analyse

- Sei  $(L,R)$  ein fester minimaler Schnitt in  $G$
- $(L,R)$  habe  $k$  kreuzende Kanten, Menge  $C$
- $G$  hat mindestens  $kn/2$  Kanten!
- Uns interessiert die Wahrscheinlichkeit, dass keine kreuzende Kante in  $C$  je gewählt wird
  - Dann ist am Ende  $(L,R)$  der erreichte Schnitt
- $E_i$  sei das Ereignis, dass eine Kante in  $C$  in Iteration  $i$  gewählt wird
  - Schritt 1:  $\text{Prob}(E_1) \leq k/(kn/2) = 2/n$
- Angenommen,  $E_1$  ist nicht eingetreten, dann ist der minimale Schnitt immer noch  $k$  und es gibt  $n-1$  Knoten, also  $(n-1)k/2$  Kanten:
  - Schritt 2:  $\text{Prob}(E_2 | \neg E_1) \leq 2/(n-1)$



# Analyse

- Per Induktion:
  - $\text{Prob}(E_i | \neg E_1, \dots, \neg E_{i-1}) \leq 2/(n-i+1)$
  - Für  $i=n-2$ :  $\text{Prob} \leq 2/3$
- Allgemein:  
 $\text{Prob}(\neg E_1 \text{ und } \neg E_2 \text{ und } \dots \text{ und } \neg E_{n-2})$   
 $= \prod_i \text{Prob}(\neg E_i | \neg E_1 \text{ und } \dots \text{ und } \neg E_{i-1})$
- Daher  $\text{Prob}(\text{Keine Kante in } C \text{ wird kontrahiert})$   
 $\geq \prod (1 - 2/(n-i+1)) \geq 2/(n(n-1))$
- Also insgesamt erhalten wir den Schnitt  $C$  mit Wahrscheinlichkeit  $\Omega(1/n^2)$
- Gesamtprozedur heie Contract
- Wir wiederholen Contract  $n^2$  mal und erhalten einen minimalen Schnitt mit Wahrscheinlichkeit

$$\gg \left(1 - \left(1 - \frac{2}{n \cdot (n-1)}\right)^{n^2}\right) \approx 1 - 1/e^2$$



# Implementierung

- Eine Iteration (Contract) kann in Zeit  $O(n^2)$  ausgeführt werden
- So ergibt sich Laufzeit  $O(n^4)$
  
- Idee zur Verbesserung:
  - Auftretende Fehlerwahrscheinlichkeiten sind sehr unbalanciert, von  $2/n$  bis  $2/3$
  - Verwende „genaueren“ Algorithmus für Min Cut nach einigen Kontraktionen?
  - Idee: Verwende zwei unabhängige Läufe der Contract Subroutine, allerdings jeweils nur, bis die Knotenzahl auf einen Wert  $t$  reduziert ist
  - Contract( $t$ ): Verschmelze die Knoten für eine uniform zufällige Kante, bis nur noch  $t$  Knoten vorhanden



# Algorithmus Fast Cut

- Verwendet Prozedur Contract
- Eingabe: Multigraph  $G$
- Ausgabe: Schnitt
  - Für  $n \leq 6$  verwende brute force
  - Setze  $t = \lceil 1 + n / \sqrt{2} \rceil$
  - Benutze Contract( $t$ ) Iterationen zweimal unabhängig voneinander, um Graphen  $H_1$  und  $H_2$  zu bestimmen, mit  $t$  Knoten
  - Berechne min Cuts in  $H_1$  und  $H_2$  rekursiv
  - Ausgabe: kleinerer Cut der beiden