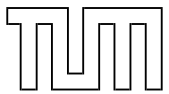
Fortgeschrittene Netzwerk- und Graph-Algorithmen

Prof. Dr. Hanjo Täubig

Lehrstuhl für Effiziente Algorithmen
(Prof. Dr. Ernst W. Mayr)
Institut für Informatik
Technische Universität München

Wintersemester 2010/11



Versteckte Cliquen

Folgerung

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es keinen Polynomialzeit-Algorithmus, der eine Clique der Größe k in einem Graphen findet, der garantiert eine solche Clique der Größe k enthält.

Bemerkung:

Die Schwierigkeit, eine versteckte Clique zu finden, hängt nicht von der Größe der Clique ab.

Auch Cliquen der Größe $(1 - \varepsilon) \cdot n$ können nicht in Polynomialzeit gefunden werden.

Maximum-Clique: besserer exponentieller Algorithmen

Satz

Eine Maximum-Clique (also eine Clique maximaler Kardinalität) kann in Zeit $\mathcal{O}^*(1.3803^n)$ berechnet werden.

($\mathcal{O}^*$ ignoriert polynomielle Faktoren)

Maximum-Clique: besserer exponentieller Algorithmus

Beweis.

- Falls $\delta(G) \geq n - 3$, dann
  - fehlen in G nur einfache Pfade und Kreise im Vergleich zum vollständigen Graphen K_n

  $\Rightarrow$ Maximum Clique kann in $\mathcal{O}(m + n)$ berechnet werden:
  - betrachte zur Veranschaulichung den komplementären Graphen

  $\Rightarrow$ Cliquen entsprechen unabhängigen Mengen
  - In einem Pfad P kann man die Größe einer unabhängigen Menge (independent set) maximaler Kardinalität berechnen als $\lceil |V(P)|/2 \rceil$.
  - In einem Kreis C ist die Größe $\lfloor |V(C)|/2 \rfloor$.
  - Da die Pfade und Kreise paarweise disjunkt sind, kann man die einzelnen Werte einfach addieren.

Maximum-Clique: besserer exponentieller Algorithmus

Beweis.

- Ansonsten sei v ein Knoten mit Grad $\deg_G(v) \leq n-4$
- Jede Maximum-Clique ist entweder
  - $\{v\}$ vereinigt mit einer Maximum-Clique von $G[N(v)]$ oder
  - eine Maximum-Clique von $G[V \setminus \{v\}]$.

$\Rightarrow$ Rekursive Berechnung mit worst-case-Zeit

$$T(n) \leq T(n-4) + T(n-1) + c \cdot (m+n)$$

$\Rightarrow$ mit Erzeugendenfunktionen kann man zeigen, dass $T(n) \in \mathcal{O}^*(\beta^n)$ mit $\beta \approx 1.3803$, wobei β die größte reelle Nullstelle des charakteristischen Polynoms $\beta^4 - \beta^3 - 1$ ist

Approximation von Maximum-Cliquen

Approximation der größten Clique mit Faktor $n/2$ ist einfach: wähle die Endknoten einer Kante, falls es eine gibt.

Satz

Es gibt einen Algorithmus, der bei Eingabe eines Graphen G mit n Knoten in polynomieller Zeit eine Clique ausgibt, deren Größe maximal um einen Faktor $\mathcal{O}\left(\frac{n}{(\log n)^2}\right)$ von der Maximum-Cliquengröße $\omega(G)$ abweicht.

Satz

Falls nicht $\mathcal{NP} = \mathcal{ZPP}$ gilt, dann existiert kein Polynomialzeitalgorithmus, der bei Eingabe eines Graphen G mit n Knoten eine Clique ausgibt, deren Größe maximal um einen Faktor $n^{1-\epsilon}$ von der Maximum-Cliquengröße $\omega(G)$ abweicht (für jedes $\epsilon > 0$).

$\mathcal{ZPP}$: Klasse der Probleme, die von randomisierten Algorithmen in erwarteter Polynomialzeit gelöst werden können ohne Fehler zu machen

Suche nach Cliquen fester Größe

- In einigen Fällen reicht es, nach Cliquen fester Größe zu suchen

$\Rightarrow$ Cliquengröße wird nicht als Teil der Eingabe betrachtet

- Vollständige Suche: $\mathcal{O}(k^2 \cdot n^k) = \mathcal{O}(n^k)$,
  wenn Cliquengröße k fest ist

Bessere Komplexität für Dreiecke

- $A(G)$: Adjazenzmatrix von Graph G
- $B(G) = A(G)^2 = A(G) \cdot A(G)$: Einträge b_{ij} sind die Wege der Länge 2 zwischen den Knoten v_i und v_j

$\Rightarrow$ läßt sich durch schnellere Matrixmultiplikation ausrechnen, z.B. in $\mathcal{O}(n^{2.376})$ (Coppersmith / Winograd, 1990)

- Existiert ein Eintrag $b_{ij} \geq 1$ mit $i \neq j$, dann gibt es einen Knoten $u \in V$, der zu v_i und zu v_j adjazent ist.
- Wenn nun auch eine Kante $\{v_i, v_j\}$ existiert, dann enthält der Graph ein Dreieck $\{v_i, v_j, u\}$

$\Rightarrow$ Checke für alle echt positiven Werte b_{ij}, ob es eine Kante $\{v_i, v_j\}$ gibt

$\Rightarrow$ Zeit-Komplexität: $\mathcal{O}(n^\alpha)$, wobei $\alpha < 2.376$ der Exponent für Matrixmultiplikation ist

Cliquen fester Größe $k \geq 3$

Sei $\alpha(r, s, t)$ so definiert, dass man die Multiplikation einer $n^r \times n^s$-Matrix mit einer $n^s \times n^t$-Matrix in Zeit $\mathcal{O}\left(n^{\alpha(r,s,t)}\right)$ berechnen kann.

Satz

Für jedes $k \geq 3$ existiert ein Algorithmus, der eine Clique der Größe k in einem Graphen mit n Knoten finden kann (falls eine solche existiert) und der in Zeit $\mathcal{O}\left(n^{\beta(k)}\right)$ läuft, wobei $\beta(k) = \alpha(\lfloor k/3 \rfloor, \lceil (k-1)/3 \rceil, \lceil k/3 \rceil)$.

Cliquen fester Größe $k \geq 3$

Beweis.

- Seien $k_1 = \lfloor k/3 \rfloor$, $k_2 = \lceil (k-1)/3 \rceil$ und $k_3 = \lceil k/3 \rceil$
- Es gilt $k = k_1 + k_2 + k_3$.
- Konstruiere tripartiten Hilfsgraph $\tilde{G}$
  - $\tilde{V} = \tilde{V}_1 \cup \tilde{V}_2 \cup \tilde{V}_3$, wobei V_i aus allen Cliquen der Größe k_i in G besteht (kann man mit dem naiven Algorithmus in $\mathcal{O}(n^{k_i})$ berechnen)
  - Knoten $U \in \tilde{V}_i$ und $U' \in \tilde{V}_j$ sind adjazent in $\tilde{G}$ g.d.w. $i \neq j$ und $U \cup U'$ eine Clique der Größe $k_i + k_j$ in G ist.
  - Berechne dazu die partiellen Adjazenzmatrizen A_{12}, A_{13} und A_{23}, die die Kanten zwischen den jeweiligen Knotenmengen darstellen
- Teste $\tilde{G}$ auf Dreiecke

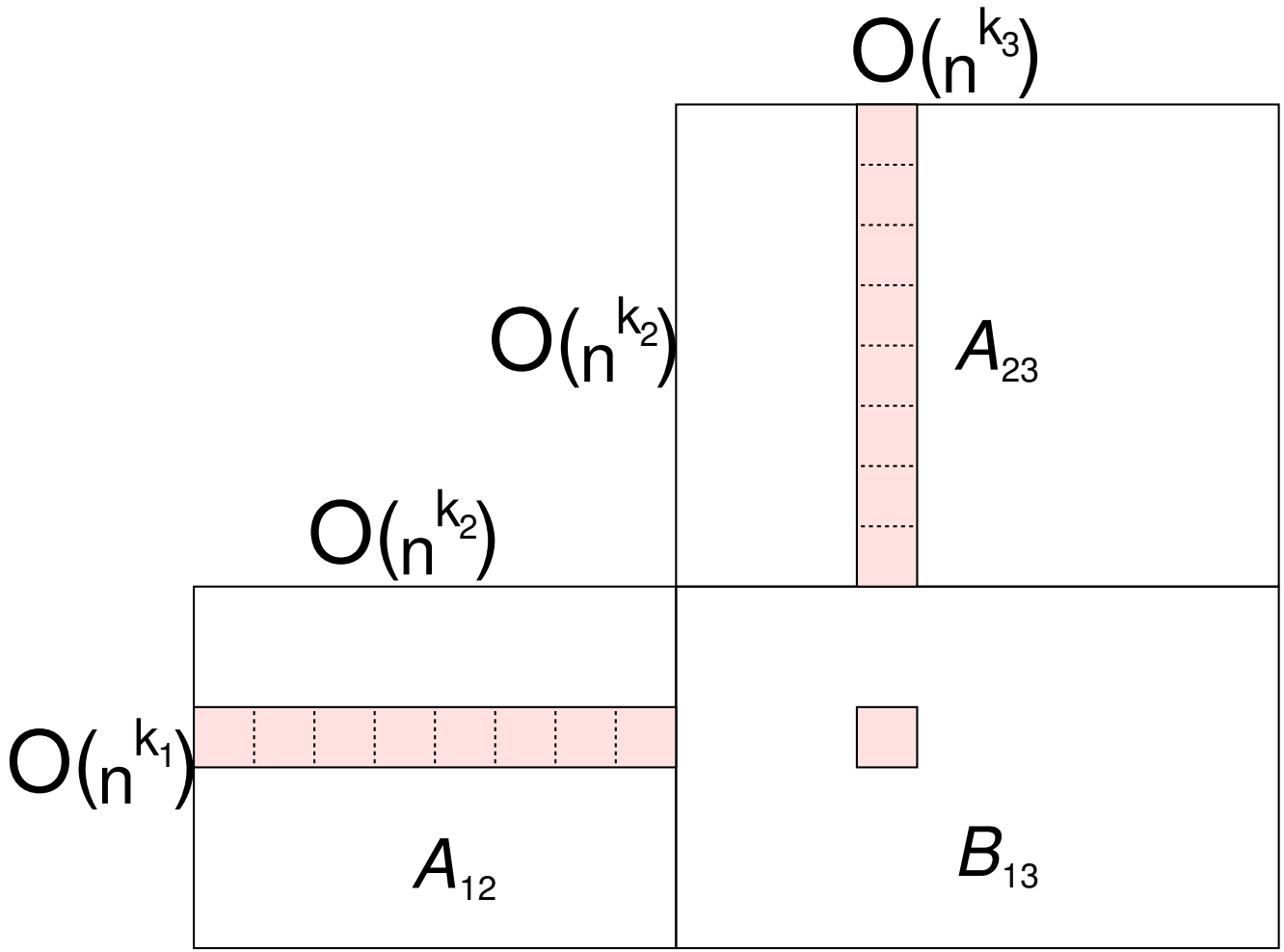
Cliquen fester Größe $k \geq 3$

Beweis.

- Dreieck $\{U_1, U_2, U_3\}$ impliziert eine Clique der Größe k in G
- geht mit schneller Matrixmultiplikation, aber hier muss eine $\mathcal{O}(n^{k_1}) \times \mathcal{O}(n^{k_2})$-Matrix (Kanten zwischen $\tilde{V}_1$ und $\tilde{V}_2$) mit einer $\mathcal{O}(n^{k_2}) \times \mathcal{O}(n^{k_3})$-Matrix (Kanten zwischen $\tilde{V}_2$ und $\tilde{V}_3$) multipliziert werden, und zwar in Zeit $\mathcal{O}\left(n^{\beta(k)}\right)$
- Berechnung der drei Matrizen A_{12}, A_{23} und A_{13} in $\mathcal{O}\left(n^{\max\{k_1+k_2, k_1+k_3, k_2+k_3\}}\right) = \mathcal{O}\left(n^{\lceil \frac{2k}{3} \rceil}\right)$

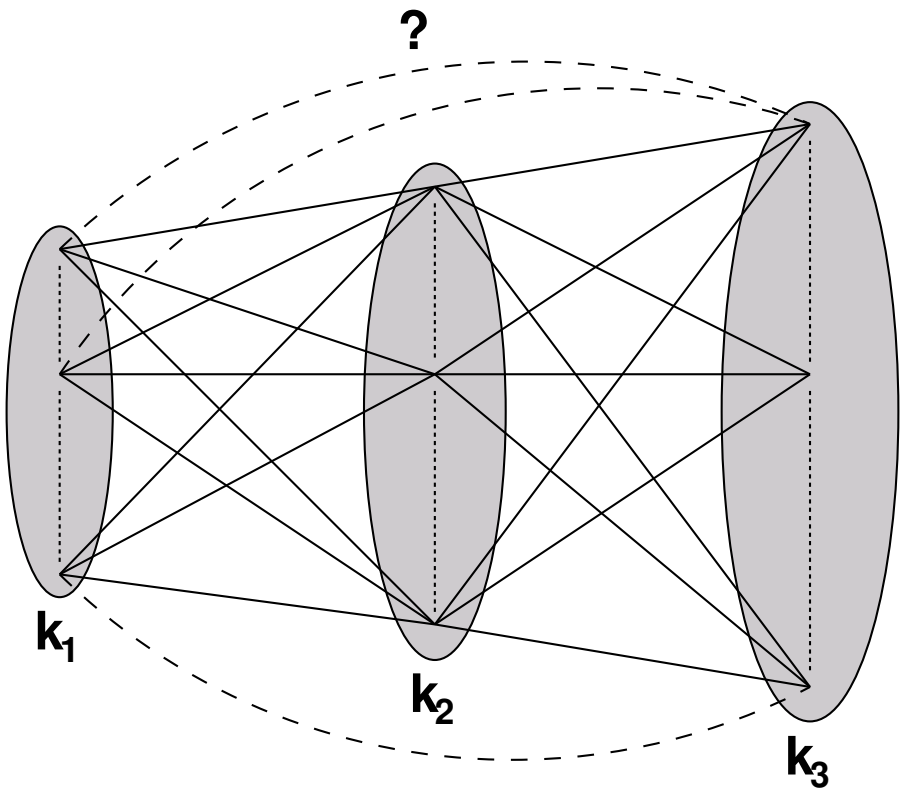
  (wird dominiert durch die Zeit $\mathcal{O}(n^{\beta(k)})$ für die Multiplikation der rechteckigen Matrizen, also $B_{13} = A_{12} \cdot A_{23}$)

Cliquen fester Größe $k \geq 3$



B_{13} wird dann mit A_{13} verknüpft

Cliquen fester Größe $k \geq 3$



Cliquen fester Größe: Beispielkomplexitäten

Cliquengröße	Vollständige Suche	Matrixmultiplikation
3	$\mathcal{O}(n^3)$	$\mathcal{O}(n^{2.376})$
4	$\mathcal{O}(n^4)$	$\mathcal{O}(n^{3.376})$
5	$\mathcal{O}(n^5)$	$\mathcal{O}(n^{4.220})$
6	$\mathcal{O}(n^6)$	$\mathcal{O}(n^{4.751})$
7	$\mathcal{O}(n^7)$	$\mathcal{O}(n^{5.751})$
8	$\mathcal{O}(n^8)$	$\mathcal{O}(n^{6.595})$

Cliquen fester Größe: Mitgliedszahlen

Satz

Für jedes $k \geq 3$ existiert ein Algorithmus, der in Zeit $\mathcal{O}\left(n^{\beta(k)}\right)$ läuft und der für jeden Knoten zählt, an wieviel Cliquen der Größe k er beteiligt ist (in einem Graphen mit n Knoten), wobei
$\beta(k) = \alpha(\lfloor k/3 \rfloor, \lceil (k-1)/3 \rceil, \lceil k/3 \rceil)$.

Cliquen fester Größe: Mitgliedszahlen

Beweis.

- Für $k = 3$ (Dreiecke) kann man nicht nur feststellen, *ob* zwei Knoten v_i und v_j zu einem Dreieck gehören, sondern auch zu wievielen.
- Wenn Kante $\{v_i, v_j\}$ in G existiert, dann ist diese Anzahl gleich dem Eintrag b_{ij} in der quadrierten Adjazenzmatrix $B(G) = A(G) \cdot A(G)$.
- Anwendung im allgemeinen Fall von $\tilde{G}$:
  für jeden Knoten $v \in V$ sei $C_k(v)$ die Anzahl verschiedener Cliquen der Größe k, in denen v enthalten ist.
- Entsprechend sei $\tilde{C}_3(U)$ die Anzahl der Dreiecke in $\tilde{G}$, zu denen Knoten U gehört.
  (U ist eine Clique der Größe kleiner als k)

Cliquen fester Größe: Mitgliedszahlen

Beweis.

- Cliquen der Größe k können viele verschiedene Repräsentationen in $\tilde{G}$ haben.
- Diese Anzahl ist die Anzahl der Partitionierungen von einer Menge der Kardinalität k in drei Mengen der Kardinalitäten k_1, k_2 und k_3, also der Multinomialkoeffizient $\binom{k}{k_1,k_2,k_3}$.
- o.B.d.A. sei k_1 der kleinste der drei Parameter.
  Sei $\mathcal{U}(v)$ die Menge aller Cliquen U der Größe k_1 in G, so dass $v \in U$. Dann gilt:

$$\sum_{U \in \mathcal{U}(v)} \tilde{C}_3(U) = \binom{(k-1)}{(k_1-1), k_2, k_3} \cdot C_k(v)$$

- Berechne linke Seite in $\mathcal{O}\left(n^{\beta(k)}\right)$ (berechne Matrizen; suche Einträge für alle U, die v enthalten); Berechne $C_k(v)$)