Fortgeschrittene Netzwerk- und Graph-Algorithmen

Dr. Hanjo Täubig

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

Wintersemester 2007/08

Übersicht

Ziel

- Algorithmus zur Lösung des All Pairs Shortest Paths (APSP) Problems
- löst APSP-Problem für gewichtete gerichtete Graphen, in denen die Kantengewichte ganze Zahlen mit kleinem Absolutbetrag sind, in Zeit $\tilde{\mathcal{O}}(n^{2+\mu})$
- μ erfüllt dabei die Bedingung $\omega(1,\mu,1) = 1 + 2\mu$, wobei $\omega(1,\mu,1)$ der Zeitkomplexitätsexponent der Multiplikation einer $n \times n^{\mu}$ mit einer $n^{\mu} \times n$ Matrix ist.
- Momentan beste Schranke: $\mu < 0,575$ (Coppersmith) impliziert Komplexität $\tilde{\mathcal{O}}(n^{2,575})$.

Distanzprodukt

Definition

Sei $A = (a_{ij})$ eine $n \times m$ Matrix und $B = (b_{ij})$ eine $m \times n$ Matrix.
Das min/plus- oder Distanzprodukt von A und B, symbolisch $A \star B$ ist die $n \times n$ Matrix $C = (c_{ij})$, so dass gilt:

$$c_{ij} = \min_{k=1}^{m} \{a_{ik} + b_{kj}\} \text{ für } 1 \leq i, j \leq n.$$

Distanzprodukt-Berechnung

- Das Distanzprodukt kann naiv in Zeit $\mathcal{O}(n^2 m)$ berechnet werden.
- Alon et al.:
  - Methode mit schneller Matrixmultiplikation und schneller Integermultiplikation (Schönhage-Strassen) für den Fall, dass die Werte im Bereich $\{-M, \ldots, 0, \ldots, M\} \cup \{+\infty\}$ liegen
  - Laufzeit $\tilde{\mathcal{O}}(Mn^{\omega(1,r,1)})$, wobei $m = n^r$.
  - $\mathcal{O}(n^{\omega(1,r,1)})$ ist hier die Anzahl algebraischer Operationen, die benötigt werden, um das (normale) Produkt einer $n \times n^r$ und einer $n^r \times n$ Matrix zu berechnen
- Laufzeit hängt von M ab. Für große Werte von M ist der naive (von M unabhängige) Algorithmus schneller.
- Der folgende Algorithmus benutzt die schnellere von beiden Methoden.

Algorithmus zur Distanzprodukt-Berechnung

Algorithmus 16 : `dist-prod`(A, B, M)

Input : $n \times m$ Matrix A, $m \times n$ Matrix B (wobei $m = n^r$). Alle Elemente von A und B sind ganze Zahlen, wobei alle Einträge mit Absolutwert $> M$ durch ∞ ersetzt werden.

Output : Distanzprodukt $C = A \star B$ ($n \times n$ Matrix)

if $Mn^{\omega(1,r,1)} \leq n^{2+r}$ **then**

$$a'_{ij} \leftarrow \begin{cases} (m+1)^{M-a_{ij}} & \text{wenn } |a_{ij}| \leq M \\ 0 & \text{sonst} \end{cases};$$

$$b'_{ij} \leftarrow \begin{cases} (m+1)^{M-b_{ij}} & \text{wenn } |b_{ij}| \leq M \\ 0 & \text{sonst} \end{cases};$$

$C' \leftarrow \mathtt{fast-prod}(A', B')$;

$$c_{ij} \leftarrow \begin{cases} 2M - \lfloor \log_{m+1} c'_{ij} \rfloor & \text{wenn } c'_{ij} > 0 \\ +\infty & \text{sonst} \end{cases};$$

else

$c_{ij} \leftarrow \min_{k=1}^{m}\{a_{ik} + b_{kj}\} \quad (1 \leq i, j \leq n)$;

return C;

Algorithmus zur Distanzprodukt-Berechnung

Lemma

Algorithmus `dist-prod` berechnet das Distanz-Produkt einer $n \times n^r$ und einer $n^r \times n$ Matrix, deren endliche Einträge alle Absolutbetrag höchstens M haben in Zeit $\tilde{\mathcal{O}}(\min\{Mn^{\omega(1,r,1)}, n^{2+r}\})$.

Beweis.

- Wenn $n^{2+r} < Mn^{\omega(1,r,1)}$, dann berechnet `dist-prod` das Distanz-Produkt von A und B mit Hilfe des naiven Algorithmus in Zeit $\mathcal{O}(n^{2+r})$ und wir sind fertig

Algorithmus zur Distanzprodukt-Berechnung

Beweis.

- Fall $n^{2+r} \geq Mn^{\omega(1,r,1)}$:
  - Korrektheit: $\forall i,j \in \{1,\ldots,n\}$:

$$c'_{ij} = \sum_{\substack{k \in \{1,\ldots m\} \\ a_{ik} \neq \infty \\ b_{kj} \neq \infty}} (m+1)^{2M-(a_{ik}+b_{kj})}$$

$\Rightarrow$

$$c_{ij} = \min_{k=1}^{m}\{a_{ik} + b_{kj}\} = 2M - \lfloor \log_{m+1} c'_{ij} \rfloor$$

Algorithmus zur Distanzprodukt-Berechnung

Beweis.

- weiter Fall $n^{2+r} \geq Mn^{\omega(1,r,1)}$:
  - Komplexität?
  - `fast-prod` führt $\tilde{\mathcal{O}}(n^{\omega(1,r,1)})$ arithmetische Operationen auf $\mathcal{O}(M \log n)$-Bit Integern aus.
  - Um große Zwischenergebnisse zu vermeiden, werden diese Multiplikationen z.B. $\mod (m+1)^{4M+1}$ ausgeführt.
  - Der Schönhage-Strassen-Algorithmus multipliziert zwei k-Bit Integer mit $\mathcal{O}(k \log k \log \log k)$ Bit-Operationen.
  - Mit $k = \mathcal{O}(M \log n)$ erhält man für jede arithmetische Operation $\tilde{\mathcal{O}}(M \log n)$.
  - Die Logarithmen in der Berechnung von c_{ij} können mit binärer Suche implementiert werden.
  - Die Komplexität ist also $\tilde{\mathcal{O}}(Mn^{\omega(1,r,1)})$.

□

Zeugen für Distanz-Produkte

Definition

Sei A eine $n \times m$ Matrix und B eine $m \times n$ Matrix.
Eine $n \times n$ Matrix heißt Zeugen-Matrix für das Distanz-Produkt $C = A \star B$, wenn für alle $1 \leq i, j \leq n$ gilt, dass $1 \leq w_{ij} \leq m$ und $c_{ij} = a_{i,w_{ij}} + b_{w_{ij},j}$.

- Man kann den Algorithmus `dist-prod` so erweitern, dass er auch eine Zeugen-Matrix zurückliefert. Die Laufzeit erhöht sich dabei nur um einen polylogarithmischen Faktor.

Zeugen für Distanz-Produkte

- Eine einfache (aber teure) Möglichkeit, die Zeugen für $C = A \star B$ zu berechnen, ist:
  - A ist $n \times m$, B ist $m \times n$ Matrix
  - Seien $A' = (a'_{ij})$ und $B' = (b'_{ij})$ Matrizen, so dass gilt: $a'_{ij} = ma_{ij} + j - 1$ und $b'_{ji} = mb_{ji}$ für alle $1 \leq i \leq n$ und $1 \leq j \leq m$.
  - Wenn man nun das Distanzprodukt $C' = A' \star B'$ berechnet, dann ist $\lfloor C'/m \rfloor$ das Distanzprodukt $A \star B$ und $(C' \bmod m) + 1$ ist eine korrespondierende Zeugen-Matrix.
  - Außerdem sind alle Zeugen die kleinsten möglichen Zeugen.
  - Der Nachteil ist: die Einträge von A und B werden mit m multipliziert, was eine Verlangsamung der Prozedur `dist-prod` um den Faktor m zur Folge hat (der sehr groß sein kann).

Zeugen für Distanz-Produkte

- Es gibt einen effizienteren Weg, die Zeugen zu finden.
- Zunächst für den Fall eindeutiger Zeugen:
  - Für $1 \leq k \leq m$ und $1 \leq \ell \leq \lceil \log_2 m \rceil + 1$ sei $\mathrm{bit}_\ell(k)$ das ℓ-te Bit in der Binärrepräsentation von k (wobei $\mathrm{bit}_1(k)$ das Least Significant Bit von k ist).
  - Sei $I_\ell = \{k : \quad 1 \leq k \leq m \quad \wedge \quad \mathrm{bit}_\ell(k) = 1\}$.

Definition

Sei A eine $n \times m$ Matrix und sei $I \subseteq \{1, 2, \ldots, m\}$ eine Teilmenge der (Spalten-)Indizes.
Dann ist $A[*, I]$ definiert als die Matrix, die aus den Spalten von A besteht, deren Index zu I gehört.
Entsprechend ist für eine $m \times n$ Matrix B der Ausdruck $B[I, *]$ gleichbedeutend mit der Matrix, die sich aus den Zeilen von B zusammensetzt, deren Index zu I gehört.

Zeugen für Distanz-Produkte

- Fortsetzung für den Fall eindeutiger Zeugen:
  - Um alle eindeutigen Zeugen für Elemente von $C = A \star B$ zu finden, berechnet man die $\mathcal{O}(\log m)$ Distanzprodukte $C_\ell = A[*, I_\ell] \star B[I_\ell, *]$ für $1 \leq \ell \leq \lceil \log_2 m \rceil + 1$.
  - Sei $C_\ell = (c_{ij}^{(\ell)})$.
  - $c_{ij}^{(\ell)} = c_{ij}$ gilt genau dann, wenn es einen Zeugen für c_{ij} gibt, dessen ℓ-tes Bit 1 ist.
  - Falls c_{ij} einen eindeutigen Zeugen hat, dann können diese Bedingungen benutzt werden, um die einzelnen Bits in der Binärrepräsentation von w_{ij} (und damit den Wert w_{ij}) zu bestimmen.
  - Man beachte, dass man nicht wissen muss, ob c_{ij} einen eindeutigen Zeugen hat. Man rekonstruiert einen Kandidaten w_{ij} und überprüft, ob $c_{ij} = a_{i,w_{ij}} + b_{w_{ij},j}$ gilt.