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

1. Graph-Algorithmen mit Matrixmultiplikation
   - All Pairs Shortest Paths

Zeugen für Distanz-Produkte

- Vorgehen im Fall mehrerer Zeugen: Sampling
  - Intention:
    Wir wollen eine Teilmenge der Indizes $\{1, \ldots, m\}$ auswählen, so dass *genau ein* Index ein Zeuge für ein bestimmtes c_{ij} ist.
  - Problem: Wir wissen nicht wieviele Zeugen es gibt.
    - Wenn es wenige Zeugen gibt, brauchen wir eine große Teilmenge der Indizes, damit überhaupt ein Zeuge dabei ist.
    - Wenn es viele gibt, dürfen wir nur eine kleine Teilmenge wählen, da die Auswahl von mehr als einem Zeugen dazu führt, dass wir die Binärrepräsentation nicht rekonstruieren können.
  - Lösung:
    Wir probieren alle möglichen Größenordnungen 2^r (mit $r \in \{1, \ldots, \log m\}$) für die Anzahl der Zeugen aus.
  - Für jedes r wählt man $s = c \log n$ zufällige Teilmengen $R_{r1}, \ldots, R_{rs}$ der Größe $m/2^r$ aus der Grundmenge $\{1, \ldots, m\}$.

Zeugen für Distanz-Produkte

- Vorgehen im Fall mehrerer Zeugen (Fortsetzung):
  - Für jede solche zufällige Teilmenge R_{rt} (wobei $1 \leq r \leq \log m$ und $1 \leq t \leq s$) suchen wir eindeutige Zeugen für das Produkt $A[*, R_{rt}] \star B[R_{rt}, *]$.
  - Wenn solch ein Zeuge gefunden wird, überprüft man, ob er auch ein Zeuge für das ursprüngliche Distanzprodukt $A \star B$ ist. (Durch das Weglassen von Zeilen/Spalten könnte das urprüngliche Minimum entfernt worden sein.)
  - Wenn die Konstante c groß genug ist, dann werden auf diese Weise mit sehr hoher Wahrscheinlichkeit Zeugen für alle Positionen gefunden (hier ohne Beweis).
  - Dieses Vorgehen führt zu einem randomisierten Algorithmus für die Berechnung der Zeugen des Distanzprodukts $A \star B$, der $\mathcal{O}(\log^3 n)$ einfache Distanzprodukte von Matrizen gleicher oder kleinerer Größe benutzt.

Zeugen für Distanz-Produkte

- Dieser Algorithmus kann mit Hilfe von Ergebnissen von Alon und Naor (1996) derandomisiert werden, wobei sich die Komplexität nur um einen polylogarithmischen Faktor verschlechtert.

Lemma

Eine erweiterte Version von Algorithmus `dist-prod` *berechnet das Distanzprodukt einer $n \times n^r$ Matrix mit einer $n^r \times n$ Matrix, deren endliche Einträge alle Betrag höchstens M haben, zusammen mit einer entsprechenden Zeugenmatrix in Zeit* $\tilde{O}(\min\{Mn^{\omega(1,r,1)}, n^{2+r}\})$.

Randomisierter Kürzeste-Wege-Algorithmus

Algorithmus 17 : `rand-short-path`(D)

Input : $n \times n$ Matrix D der Kantengewichte
Output : $n \times n$ Matrix F, die mit hoher Wahrscheinlichkeit, die Distanzen des Graphen enthält, sowie eine Zeugen-Matrix W

$F \leftarrow D$;
$W \leftarrow 0$;
$M \leftarrow \max\{|d_{ij}| : d_{ij} \neq +\infty\}$;
for $\ell \leftarrow 1 \ldots \lceil \log_{3/2} n \rceil$ **do**
  $s \leftarrow (3/2)^\ell$;
  $B \leftarrow \mathtt{rand}(\{1, \ldots, n\}, (9 \ln n)/s)$;
  $(F', W') \leftarrow \mathtt{dist} - \mathtt{prod}(F[*, B], F[B, *], sM)$;
  foreach $1 \leq i, j \leq n$ **do**
    if $f'_{ij} < f_{ij}$ **then**
      $f_{ij} \leftarrow f'_{ij}$;
      $w_{ij} \leftarrow b_{w'_{ij}}$;
return (F, W);

Randomisierter Kürzeste-Wege-Algorithmus

- Einfacher randomisierter Algorithmus `rand-short-path` zur Bestimmung der Distanzen und einer Repräsentation kürzester Pfade in einem gerichteten Graphen mit n Knoten, in dem alle Kantengewichte aus $\{-M, \ldots, 0, \ldots, M\}$ kommen
- Annahme: $V = \{1, \ldots, n\}$
- Eingabe: $n \times n$ Matrix D mit Gewichten bzw. Kantenlängen (d_{ij} ist Länge der gerichteten Kante von i nach j, sonst ∞)
- Zunächst wird F mit D initialisiert.
- Dann werden $\lceil \log_{3/2} n \rceil$ Iterationen ausgeführt, in denen s auf $(3/2)^\ell$ gesetzt wird und dann aus der Knotenmenge eine Teilmenge B ausgewählt wird, in die jeder Knoten unabhängig mit Wahrscheinlichkeit $\min\{1, (9 \ln n)/s\}$ aufgenommen wird.
- Dann wird das Distanzprodukt $F' = F[*, B] \star F[B, *]$ (mit Betragsobergrenze sM) berechnet.

Randomisierter Kürzeste-Wege-Algorithmus

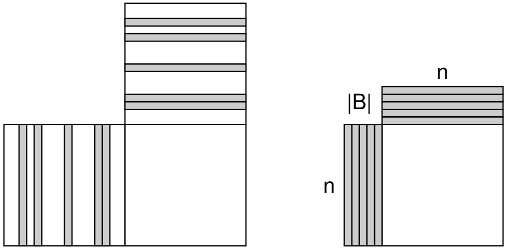

Abbildung: Ersetzung des quadratischen Produkts $F \star F$ durch das rechteckige Produkt $F[*, B] \star F[B, *]$

Randomisierter Kürzeste-Wege-Algorithmus

- Es wird dabei auch eine Zeugenmatrix W' bestimmt.
- Dann wird jeder Eintrag von F' mit dem zugehörigen Eintrag von F verglichen.
- Im Fall einer Verbesserung wird der Wert von F' nach F und der Zeuge von W' nach W übertragen.
  ($b_{w'_{ij}}$ ist das w'_{ij}-te Element von B)

Randomisierter Kürzeste-Wege-Algorithmus

- Sei $\delta(i,j)$ die (gewichtete) Distanz von i nach j.

Lemma

Zu jeder Phase gilt für Algorithmus `rand-short-path` *für alle $i,j \in V$:*

1. $f_{ij} \geq \delta(i,j)$.
2. *Falls $w_{ij} = 0$, dann ist $f_{ij} = d_{ij}$. Ansonsten gilt $1 \leq w_{ij} \leq n$ und $f_{ij} \geq f_{i,w_{ij}} + f_{w_{ij},j}$.*
3. *Falls $\delta(i,j) = \delta(i,k) + \delta(k,j)$ und falls am Anfang einer Iteration gilt $f_{ik} = \delta(i,k)$, $f_{kj} = \delta(k,j)$, $|f_{ik}|, |f_{kj}| \leq sM$ und $k \in B$, dann gilt am Ende der Iteration $f_{ij} = \delta(i,j)$.*

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

1. Die Eigenschaft gilt bei Initialisierung von F mit D. In jeder Iteration wählt der Algorithmus eine Menge B und bestimmt

$$\begin{aligned} f'_{ij} &\leftarrow \min\{f_{ik} + f_{kj} | k \in B, |f_{ik}|, |f_{kj}| \leq sM\} \\ f_{ij} &\leftarrow \min\{f_{ij}, f'_{ij}\} \end{aligned}$$

für jedes $i, j \in V$.
Für jedes k gilt $f_{ik} + f_{kj} \geq \delta(i,k) + \delta(k,j) \geq \delta(i,j)$
(aus Induktionsvoraussetzung und Dreiecksungleichung)
Also ist der neue Wert von f_{ij} wieder eine obere Schranke für $\delta(i,j)$.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

2. Zunächst gilt für alle $i, j \in V$: $f_{ij} = d_{ij}$ und $w_{ij} = 0$ und damit ist die Aussage wahr.
   Wenn f_{ij} ein neuer Wert zugewiesen wird, gilt $1 \leq w_{ij} \leq n$ und $f_{ij} = f_{i,w_{ij}} + f_{w_{ij},j}$.
   Bis zur nächsten Wertzuweisung an f_{ij} wissen wir also, dass gilt $f_{ij} \geq f_{i,w_{ij}} + f_{w_{ij},j}$, da der Wert von f_{ij} sich nicht ändert und die Werte von $f_{i,w_{ij}}$ und $f_{w_{ij},j}$ höchstens kleiner werden können.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

3. Falls die Bedigungen wie angegeben gelten, gilt am Ende der Iteration:

$$f_{ij} \leq f'_{ij} \leq f_{ik} + f_{kj} = \delta(i,k) + \delta(k,j) = \delta(i,j)$$

Da aus der ersten Eigenschaft folgt, dass $f_{ij} \geq \delta(i,j)$, gilt somit $f_{ij} = \delta(i,j)$.

□

Randomisierter Kürzeste-Wege-Algorithmus

Lemma

Sei $s = (3/2)^\ell$ für ein $\ell \in \{1, \ldots, \lceil \log_{3/2} n \rceil\}$.
Dann gilt mit sehr hoher Wahrscheinlichkeit: Wenn es im Graphen einen kürzesten Pfad von i nach j gibt, der höchstens s Kanten benutzt, dann gilt am Ende der Iteration ℓ: $f_{ij} = \delta(i, j)$.

Beweis.

- durch Induktion über ℓ
- Induktionsanfang: Für $\ell = 1$ gilt die Behauptung.
- Induktionsschritt:
  Annahme, die Behauptung gilt für $\ell - 1$,
  zu zeigen: Behauptung gilt auch für ℓ

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

- Seien i und j zwei Knoten, die durch einen kürzesten Pfad p mit höchstens $(3/2)^\ell$ Kanten verbunden sind.
- Falls p höchstens $2s/3$ Kanten hat, dann gilt nach Induktionsvoraussetzung, dass schon nach der $(\ell - 1)$-ten Iteration gilt: $f_{ij} = \delta(i, j)$ (mit sehr hoher Wahrscheinlichkeit)
- Also Annahme: p hat mindestens $2s/3$ und höchstens $s = (3/2)^\ell$ Kanten.
- Zunächst zusätzliche Annahme: s ist durch 3 teilbar.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

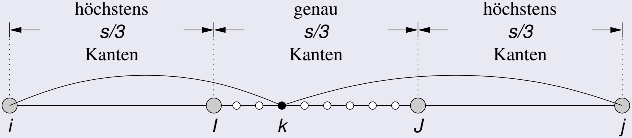

Abbildung: Korrektheitsbeweis für `rand-short-path`

- Seien I und J zwei Knoten auf dem Pfad p, die (auf p) durch exakt $s/3$ Kanten getrennt sind (Solche Knoten muss es aufgrund der Längenbeschränkung $[2s/3, s]$ immer geben).
- Sei A die Menge der Knoten von I bis J (einschließlich) auf p und sei $k \in A$.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

- Da k auf einem kürzesten Pfad von i nach j liegt, gilt $\delta(i,j) = \delta(i,k) + \delta(k,j)$.
- Da k zwischen I und J liegt, gibt es kürzeste Pfade von i nach k und von k nach j, die höchstens $2s/3$ Kanten benutzen.
- Aufgrund der Induktionsvoraussetzung wissen wir, dass zu Beginn der ℓ-ten Iteration mit sehr großer Wahrscheinlichkeit gilt: $f_{ik} = \delta(i,k)$ und $f_{kj} = \delta(k,j)$.
- Aus dem dritten Punkt des vorhergehenden Lemmas folgt daher: Falls ein $k \in A \cap B$ existiert (wobei B die gewählte Knotenmenge aus der ℓ-ten Iteration ist), dann gilt am Ende der ℓ-ten Iteration $f_{ij} = \delta(i,j)$.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

- Aber wie groß ist die Wahrscheinlichkeit, dass es solch ein $k \in A \cap B$ gibt, dass also $A \cap B \neq \emptyset$ gilt?
- Sei nun $p = \min\{1, (9 \ln n)/s\}$.
- Falls $p = 1$, dann ist $B = V$ und $A \cap B \neq \emptyset$.
- Sei also $p = (9 \ln n)/s < 1$ und jeder Knoten gehört unabhängig mit Wahrscheinlichkeit p zur Teilmenge B.
- Da $|A| \geq s/3$ ist, gilt (mit Hilfe von $1 - x \leq e^{-x}$)

$$\Pr[A \cap B = \emptyset] \leq \left(1 - \frac{9 \ln n}{s}\right)^{s/3} \leq e^{-\frac{9 \ln n}{s} \cdot \frac{s}{3}} = n^{-3}$$

- Da es weniger als n^2 Knotenpaare im Graphen gibt, ist die Fehlerwahrscheinlichkeit für den gesamten Algorithmus kleiner als $n^2 \cdot n^{-3} = 1/n$.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

Fix für s ist nicht durch 3 teilbar:

- Definiere die Sequenz $s_0 = 1$ und $s_\ell = \lceil 3s_{\ell-1}/2 \rceil$ für $\ell > 0$.
- Es gilt $s_\ell \geq (3/2)^\ell$.
- Beweis durch Induktion über ℓ, dass (mit hoher Wahrscheinlichkeit) für alle $i, j \in V$ gilt:
  Wenn es einen kürzesten $i \rightarrow j$-Pfad mit $\leq s_\ell$ Kanten gibt, dann gilt am Ende der ℓ-ten Iteration: $f_{ij} = \delta(i, j)$.
- Wenn p ein kürzester $i \rightarrow j$-Pfad mit $\leq s_\ell$ Kanten ist, betrachte Knoten I und J auf p, so dass I und J durch genau $\lfloor s_\ell/2 \rfloor$ Kanten getrennt werden und dass zwischen i und I sowie zwischen J und j höchstens $\lceil s_\ell/2 \rceil$ Kanten liegen.
- Die gleichen Argumente wie im vereinfachten Fall führen zum Beweisabschluss.

$\square$

Randomisierter Kürzeste-Wege-Algorithmus

- Jedes Paar von Knoten in einem Graphen mit n Knoten (und ohne negative Kreise) ist durch einen kürzesten Pfad mit weniger als n Kanten verbunden (falls ein Pfad existiert).
- Zusammen mit den beiden vorangegangenen Lemmas folgt daraus, dass F nach der letzten Iteration mit sehr hoher Wahrscheinlichkeit die Distanzmatrix ist.
- Weiterhin ist $\delta(i,j) = d_{ij}$ oder w_{ij} liegt auf einem kürzesten Pfad von i nach j.

Randomisierter Kürzeste-Wege-Algorithmus

Lemma

Wenn es keine negativen Kreise im Graph gibt, dann gilt nach der letzten Iteration von `rand-short-path` mit sehr hoher Wahrscheinlichkeit: $\forall i, j \in V$

1. $f_{ij} = \delta(i,j)$
2. *Falls* $w_{ij} = 0$, *dann gilt* $\delta(i,j) = d_{ij}$.
   Sonst gilt $1 \leq w_{ij} \leq n$ *und* $\delta(i,j) = \delta(i, w_{ij}) + \delta(w_{ij}, j)$.

Randomisierter Kürzeste-Wege-Algorithmus

Beweis.

1. Folgt aus
   - dem letzten Lemma,
   - dem Fakt, dass in der letzten Iteration gilt $s \geq n$,
   - dem Fakt, dass falls $\delta(i,j) < +\infty$ und falls keine negativen Kreise existieren, es einen kürzesten $i \to j$-Pfad mit höchstens $n-1$ Kanten gibt.
2. Sei nun $f_{ij} = \delta(i,j) < d_{ij}$. Aus dem zweiten Punkt des vorletzten Lemmas folgt, dass nach der letzten Iteration gilt:
   $1 \leq w_{ij} \leq n$ und $f_{ij} \geq f_{i,w_{ij}} + f_{w_{ij},j}$ bzw.
   $\delta(i,j) \geq \delta(i,w_{ij}) + \delta(w_{ij},j)$.
   Aus der Dreiecksungleichung folgt:
   $\delta(i,j) \leq \delta(i,w_{ij}) + \delta(w_{ij},j)$.
   Es gilt also $\delta(i,j) = \delta(i,w_{ij}) + \delta(w_{ij},j)$.

□

Randomisierter Kürzeste-Wege-Algorithmus

- Man kann auch leicht sehen, dass der Graph genau dann einen Kreis negativer Länge enthält, wenn ein $i \in \{1, \ldots, n\}$ mit $f_{ii} < 0$ existiert.
- Wenn es einen Pfad von i nach j gibt, der durch ein Knoten eines negativen Kreises geht, definieren wir die Distanz von i nach j als $-\infty$.
- Man kann alle solchen Paare in Zeit $\tilde{O}(n^{\omega})$ bestimmen (Galil/Margalit).
- Die von `rand-short-path` zurückgegebene Matrix W enthält eine kompakte Repräsentation von kürzesten Pfaden zwischen allen Knotenpaaren des Graphen.
- Methoden zur Rekonstruktion der kürzesten Pfade werden später beschrieben.

Randomisierter Kürzeste-Wege-Algorithmus

Komplexität von `rand-short-path`:

- Die Zeit der ℓ-ten Iteration wird dominiert durch die Berechnung des Distanzprodukt `dist-prod` einer $n \times m$ mit einer $m \times n$ Matrix, wobei $m \in \mathcal{O}((n \log n)/s)$, mit Betragsbeschränkung sM.
- Eigentlich ist m eine binomialverteilte Zufallsvariable mit $\mathbb{E}[m] \in \mathcal{O}((n \log n)/s)$, was aber an der Analyse nichts ändert.
- Unter der Annahme, dass $s = n^{1-r}$ und $M = n^t$ ist die Zeit für eine Iteration $\tilde{\mathcal{O}}(\min\{n^{t+\omega(1,r,1)+(1-r)}, n^{2+r}\})$.
- Laufzeit einer Iteration ist maximal, wenn $t + \omega(1, r, 1) + (1 - r) = 2 + r$, bzw. wenn $\omega(1, r, 1) = 1 + 2r - t$.
- Da es nur $\mathcal{O}(\log n)$ Iterationen gibt, gilt der folgende Satz.

Randomisierter Kürzeste-Wege-Algorithmus

Satz

Algorithmus `rand-short-path` findet mit sehr hoher Wahrscheinlichkeit alle Distanzen im gegebenen Graph, sowie eine kompakte Repräsentation von kürzesten Pfaden zwischen allen Knotenpaaren.
Wenn der Graph n Knoten hat und die Gewichte alle Integers mit Betrag höchstens $M = n^t$ (mit $t \leq 3 - \omega$) sind, dann ist die Laufzeit $\tilde{O}(n^{2+\mu(t)})$, wobei für $\mu = \mu(t)$ gilt:
$\omega(1, \mu, 1) = 1 + 2\mu - t$.

Randomisierter Kürzeste-Wege-Algorithmus

Bemerkungen:

- Der Ausdruck "mit sehr hoher Wahrscheinlichkeit" im Satz meint eine Wahrscheinlichkeit von mindestens $1 - 1/n$.
- Der Algorithmus kann so verändert werden, dass diese Wahrscheinlichkeit mindestens $1 - n^{-c}$ für eine beliebige Konstante c ist.
- Wenn $M > n^{3-\omega}$ ist, wird keine schnelle Matrixmultiplikation benutzt und die Laufzeit ist $\tilde{O}(n^3)$.

Komplexität im Fall $M = \mathcal{O}(1)$

- Betrachte Spezialfall $M = \mathcal{O}(1)$, z.B. wenn alle Gewichte aus $\{-1, 0, 1\}$ kommen.
- Algorithmus von Alon, Galil und Margalit braucht in diesem Fall $\tilde{\mathcal{O}}(n^{(3+\omega)/2})$, also ungefähr $\mathcal{O}(n^{2.688})$.
- Laufzeit des neuen Algorithmus von Zwick $\tilde{\mathcal{O}}(n^{2+\mu})$, wobei für μ gilt: $\omega(1, \mu, 1) = 1 + 2\mu$.
- Mit der naiven Schranke $\omega(1, r, 1) \leq 2 + (\omega - 2)r$ erhält man$\mu \leq 1/4 - \omega < 0.616$.
- Mit der besseren Schranke aus dem zweiten Lemma erhält man $\mu \leq (\alpha(\omega - 1) - 1)/(\omega + 2\alpha - 4) < 0.575$.

Komplexität im Fall $M = 1$

Folgerung

Der Algorithmus `rand-short-path` *findet mit sehr großer Wahrscheinlichkeit alle Distanzen und eine kompakte Repräsentation von kürzesten Pfaden zwischen allen Knotenpaaren in einem gerichteten Graph mit n Knoten, in dem alle Kantengewichte aus $\{-1, 0, 1\}$ kommen, in Zeit $\mathcal{O}(n^{2.575})$.*