Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Graphentheoretische Konzepte und Algorithmen

Referat - Ausarbeitung

Aufgabe 3: Flußprobleme

Autor: Adrian Helberg

Referat eingereicht im Rahmen der Vorlesung
Graphentheoretische Konzepte und Algorithmen

im Studiengang Angewandte Informatik (AI)
am Department Informatik der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. C. Klauck
Abgegeben am 28. November 2019

Inhaltsverzeichnis

1 Arbeitsplan

Im Folgenden wird die Arbeitsweise und das Vorgehen zur Erarbeitung der Aufgabenstellung vorgestellt. Hierfür wird die Aufgabenstellung gezeigt, die nötige Recherche beschrieben und die detailierte Erarbeitung vorgestellt.

1.1 Kontext

1.1.1 Aufgabenstellung

Ziel der Aufgaben ist sowohl eine Implementierung zweier Algorithmen zum Finden des maximalen Durchsatzes (Flusses) als auch deren Vergleich.

Folgende Algorithmen werden bearbeitet:

I. Der Algorithmus von **Ford und Fulkerson** [1]

II. Der Algorithmus von **Edmonds und Karp** [2]

Zusatz: Es soll nicht mittels Residualnetzwerks gearbeitet werden

Weitere Vorgaben:

- Ergebnisse sind nachvollziehbar: Ausgaben in Dateien
- Berechneter Fluss ist als Attribut an Kanten zu speichern

[1] „Ford-Fulkerson“, 1956, L.R.Ford & D.R.Fulkerson
[2] „Edmonds-Karp“, 1970, Yefim Dinitz, 1972, J.Edmonds & R.Karp

- Schnittstellen:
  - fordfulkerson:fordfulkerson(< $Filename$ >,< $Quelle$ >,< $Senke$ >): [<Liste der im letzten Lauf inspizierten Ecken>]
  - fordfulkerson:fordfulkersonT(< $Graph$ >,< $Quelle$ >,< $Senke$ >): [<Liste der im letzten Lauf inspizierten Ecken>]
  - edmondskarp:edmondskarp(< $Filename$ >,< $Quelle$ >,< $Senke$ >): [<Liste der im letzten Lauf inspizierten Ecken>]
  - edmondskarp:edmondskarpT(< $Graph$ >,< $Quelle$ >,< $Senke$ >): [<Liste der im letzten Lauf inspizierten Ecken>]
- Nachweis der erwarteten Komplexität durch Laufzeitmessung
- Gegebene Graphen sind zum Test der Korrektheit anzuwenden
- Logdateien zur Zeitmessung der Algorithmen sind anzulegen
- Bildschirmausgabe des Tests der Datei aufg3test.beam ist zu protokollieren

1.1.2 Recherche

Graphen werden hier durch $G(V, E)$ mit $|V|$ Ecken und $|E|$ Kanten beschrieben.

Im Folgenden wird auf das Kapitel „Flussprobleme“ aus dem Buch zur Vorlesung eingegangen [1] (ab Seite 95).

[...] *grundsätzlich mit schwach zusammenhängenden, schlichten gerichteten Graphen* [...]

Ein Graph heißt

- **zusammenhängend**, wenn die Knoten paarweise durch eine Kantenfolge verbunden sind
- **schwach zusammenhängend**, wenn der Graph, der entsteht, wenn man jede gerichtete Kante duch eine ungerichtete Kante ersetzt, zusammenhängend ist
- **schlicht**, wenn er ungerichtet ist und weder Mehrfachkanten, noch Schleifen besitzt

[...] *jede Kante gibt die Kapazität* $c(e_{ij}) = c_{ij}$ *der Kante an* [...]. *Aus praktischen Gründen nehmen wir dabei an, dass alle* c_{ij} *rationale Zahlen sind.*

In dieser Arbeit wird im Weiteren der Begriff **Netzwerk** verwendet, sollten Graphen die genannten Eigenschaften aufweisen.

Definition 4.1 [...] *Eine Kapazität ist eine Funktion c, die jeder Kante $e_{ij} \in E$ eine positive rationale Zahl (> 0) als Kapaziät zuordnet. Ein Fluss in G von der Quelle $q = v_1$ zu der Senke $s = v_n$ ist eine Funktion f, die jeder Kante $e_{ij} \in E$ eine nicht negative rationale Zahl zuordnet* [...]

In einer Implementierung der Algorithmen kann c als Kapazität und f als Fluss für Namen des jeweile Kantenattributs gesetzt werden.
Weiter gilt die

- **Kapazitätsbeschränkung** $(e_{\mathrm{ij}} : f(e_{\mathrm{ij}} \leq c(e_{\mathrm{ij}}))$ und
- die **Flusserhaltung**
  $(\forall_{\mathrm{j}} \in \{1, \ldots n\} : \sum_{e_{\mathrm{ij}} \in O(v_{\mathrm{i}}, e_{\mathrm{ji}} \in I(v_{\mathrm{i}}))} f(e_{1\mathrm{j}}) = \sum (f(e_{\mathrm{ij}}) - f(e_{\mathrm{ji}})) = 0)$

In dieser Arbeit wird im Weiteren der Begriff **Flussnetzwerk** verwendet, sollten Graphen diese Eigenschaften aufweisen (V, E, f, c).
Der Wert des **Flusses** f mit $d = \sum_{e_{1\mathrm{j}} \in O(q)} f(e_{1\mathrm{j}}) = \sum_{e_{\mathrm{in}} \in I(s)} f(e_{\mathrm{in}})$ beträgt in den Ecken q (Quelle) und s (Senke) 0 [Null] (**Flusserhaltung**). Dies ist bei einer Implementierung zu beachten. Die maximale Menge, die von der Quelle zur Senke transportiert werden kann ist ein Fluss maximaler Stärke.

Definition 4.2 *Ein Schnitt ist die Menge von Kanten $A(X, \bar{X})$, wobei $q \in C$ und $s \in \bar{X}$.*

Definition 4.3 *Ein Fluss, dessen Wert $min\{c(X, \bar{X}) \| A(X, \bar{X})$ ist ein beliebiger Schnitt$\}$ entspricht heißt ein **maximaler Fluss**.*

Definition 4.4 *Ein ungerichteter Weg von der Quelle q zur Senke s heißt ein vergrüßernder Weg, wenn gilt:*

- Für jede Kante e_{ij} , die auf dem Weg entsprechend ihrer Richtung durchlaufen wird (sie wird als **Vorwärtskante** bezeichnet), ist $f(e_{\mathrm{ij}}) < c(e_{\mathrm{ij}})$.
- Für jede Kante e_{ij} , die auf dem Weg entgegen ihrer Richtung durchlaufen wird (sie wird als **Rückwärtskante** bezeichnet), ist $f(e_{\mathrm{ij}}) > 0$).

Satz 4.2 *Wenn in einem Graphen G ein Fluss der Stärke d von der Quelle q zur Senke s fließt, gilt genau eine der beiden Aussagen:*

1. *Es gibt einen vergrößernden Weg.*
2. *Es gibt einen Schnitt $A(X, \bar{X})$ mit $c(X, \bar{X}) = d$.*

1. kann bei einer Implementierung eine Rekursion auslsen („Ein maximaler Fluss ist noch nicht gefunden“ - Erweiterbarkeit ist gegeben), wobei *2.* die Abbruchbedingung beschreibt („Ein maximaler Fluss ist gefunden“ - Erweiterbarkeit nicht gegeben).

1.1.3 Erarbeitung der Algorithmen

Ford-Fulkerson

Ecken des Graphen werden während der Suche nach einem vergrößernden (vergr.) Weg mit $(Vorg_\mathrm{i}, \delta_\mathrm{i})$ markiert.

- $Vorg_\mathrm{i}$ beschreibt den Vorgänger von v_i auf einem vergr. Weg
- δ_i gibt die bisher auf dem vergr. Weg maximal mögliche Änderung der Flussstärke an
- $Vorg_\mathrm{i}$ wird mit einem Vorzeichen versehen, das angibt, ob eine Kante entsprechend $(+)$ oder entgegen $(-)$ ihrer Richtung durchlaufen wird

Pseudocode:

1. *(Initialisierung)*
   Weise allen Kanten $f(e_{ij})$ als einen (initialen) Wert zu, der die Nebenbedingungen erfüllt. Markiere q mit $(undefiniert, \infty)$.

2. *(Inpektion und Markierung)*

   (a) *Falls alle markierten Ecken inspiziert wurden, gehe nach 4.*

   (b) *Wähle eine beliebige markierte, aber noch nicht inspizierte Ecke v_i und inspiziere sie wie folg (Berechnung des Inkrements)*

   - *(Vorwärtskante) Für jede Kante $e_{ij} \in O(v_i)$ mit unmarkierter Ecke v_j und $f(e_{ij})$ markiere v_j mit $(+v_i, j)$, wobei δ_j die kleinere der beiden Zahlen $c(e_{ij}) - f(e_{ij})$ und δ_i ist.*
   - *(Rückwärtskante) Für jede Kante $e_{ji} \in I(v_i)$ mit unmarkierter Ecke v_j und $f(e_{ji}) > 0$ markiere v_j mit (v_i, δ_j), wobei δ_j die kleinere der beiden Zahlen $f(e_{ji})$ und δ_i ist.*

   (c) *Falls s markiert ist, gehe zu 3., sonst zu 2.(a).*

3. *(Vergrößerung der Flussstärke)*
   Bei s beginnend lässt sich anhand der Markierungen der gefundene verg. Weg bis zur Ecke q rückwärts durchlaufen. Für jede Verwärtskante wird $f(e_{ij})$ um δ_s erhöht, und für jede Rückwärtskante wird $f(e_{ji})$ um δ_s vermindert. Anschließend werden bei allen Ecken mit Ausnahme von q die Markierungen entfernt. Gehe zu 2.

4. *Es gibt keinen verg. Weg. Der jetzige Wert von d ist optimal. Ein Schnitt $A(X, \bar{X})$ mit $c(X, \bar{X}) = d$ wird gebildet von genau denjenigen Kanten, bei denen entweder die Anfangsecke oder die Endecke inspiziert ist.*

In Schritt *1.* wird i.allg. $f(e_{\mathrm{ij}}) := 0$ für alle i und j gewählt.

1.2 Ziele

Entwurf

1.3 Diskussion

Was kann diskutiert werden?
Fragen

2 Entwurf

3 Quellen

Literatur

[1] Christoph Klauck & Christoph Maas. *Graphentheorie für Studierende der Informatik.* HAW Hamburg, 6. edition, 2015.

4 Erklärung zur schriftlichen Ausarbeitung

Hiermit erkläre ich, dass ich diese schriftliche Ausarbeitung meines Referates selbstständig und ohne fremde Hilfe verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe sowie die aus fremden Quellen (dazu zählen auch Internetquellen) direkt oder indirekt übernommenen Gedanken oder Wortlaute als solche kenntlich gemacht habe. Zudem erkläre ich, dass der zugehörige Programmcode von mir selbständig implementiert wurde ohne diesen oder Teile davon von Dritten im Wortlaut oder dem Sinn nach übernommen zu haben. Die Arbeit habe ich bisher keinem anderen Prüfungsamt in gleicher oder vergleichbarer Form vorgelegt. Sie wurde bisher nicht veröffentlicht.

Hamburg, den 28. November 2019 _____