Teilgraphen hoher Konnektivität*

Henning Thomas†
thomahen@student.ethz.ch

Oktober 2007

Zusammenfassung

Wir betrachten Approximationsalgorithmen für $\mathcal{NP}$-schwere Probleme, die den Zusammenhang von Graphen betreffen. Dabei beschäftigen wir uns zunächst mit dem Problem einem minimalen λ-fach kanten-/knotenzusammenhängenden Teilgraphen in einem ungerichteten Graphen zu finden. Insbesondere stellen wir für den Kantenzusammenhangsfall Polynomzeitalgorithmen für gewichtete und ungewichtete Graphen vor mit Approximationsgüten von 2 und $2 - 1/\lambda$. Anschließend geben wir einen $1,645$-Apprxoimationsalgorithmus für das Problem an, in einem gegebenen gerichteten Graphen einen minimalen stark zusammenhängenden Teilgraphen zu finden.

1 Einleitung

Wir beschäfitgen uns mit dem Zusammenhang von Graphen (siehe Anhang A). Die hier diskutierten Probleme tauchen in natürlicher Weise in Netzwerkstrukturen auf, wenn es darum geht die Kommunikationmöglichkeit (Erreichbarkeit) zwischen zwei Knotenpunkten zu einem bestimmten Grad sicherzustellen. Dabei entsprechen Knotenpunkte den Knoten eines Graphen und Kanäle bzw. Verbindungen den Kanten eines Graphen. Gerichtete Kanten stellen dabei unidirektionale, und ungerichtete Kanten bidirektionale Kanäle dar. Die Gewichtung von Kanten kann verschiedene Zwecke wie z.B. räumliche Distanz oder Bandbreite repräsentieren.

Definition 1. *Sei $\lambda \in \mathbb{N}$. Ein Graph G heißt λ-fach kanten-/knotenzusammenhängend (im Fall $\lambda = 1$ einfach zusammenhängend), falls für alle $u, v \in V(G)$ mindestens λ kanten-/knotenunabhängige Pfade von u nach v existieren.*

Äquivalent zu der Bedingung aus Definition 1 ist für $\lambda \geq 2$ die Bedingung, dass das Entfernen von $\lambda - 1$ Kanten/Knoten aus dem Graphen den Graphen zusammenhängend lässt. Man kann außerdem leicht feststellen, dass Knotenzusammenhang eine stärkere Forderung an den Graphen stellt, oder genauer: Ein λ-fach *knoten*zusammenhängender Graph G ist insbesondere auch λ-fach *kanten*zusammenhängend.
In Abschnitt 2 widmen wir uns folgendem Problem zu ungerichteten Graphen: Gegeben sei ein ungerichteter (evtl. gewichteter) λ-fach kanten-/knotenzusammenhängender Graph G. Finde einen minimalen Teilgraphen $G_0 \subseteq G$, der λ-fach kanten-/knotenzusammenhängend ist. Im allgemeinen Fall (bereits für $\lambda \geq 2$) sind diese Probleme $\mathcal{NP}$-schwer. Wir werden uns hier hauptsächlich dem einfacheren Fall des Kantenzusammenhangs widmen und zum Knotenzusammenhang nur eine kurze Anmerkung machen. Der Untersuchung des Kantenzusammenhangs können dabei gewichtete oder ungewichtete Graphen zugrunde liegen, wobei der ungewichtete Fall ein Spezialfall des gewichteten ist, nämlich mit der Einschränkung der Kantengewichte auf 1 oder ∞. Im Falle von gewichteten Graphen stellen wir einen 2-Approximationsalgorithmus aus [7] vor, während wir im ungewichteten Fall mit einem verfeinerten Linearzeit-Algorithmus (siehe [8]) eine Approximationsgüte von $2 - 1/\lambda$ erreichen.
In Abschnitt 3 untersuchen wir gerichtete Graphen. Wir betrachten das Problem, bei dem ein gerichteter, stark zusammenhängender Graph G gegeben ist, und nach einem minimalen stark zusammenhängenden

*Vortrag gehalten auf der Sommerakademie der deutschen Studienstiftung in Görlitz 2007. Es handelt sich dabei um eine Vorstellung von [5] Kap. 6

†Institut für Theoretische Informatik, ETH Zürich, 8092 Zürich, Schweiz

Teilgraphen $G_0 \subseteq G$ gefragt wird. Für dieses Problem geben wir einen Approximationsalgorithmus mit Güte $1,645$ aus [6] an.

2 Das Kantenzusammenhangsproblem

Gegeben sei eine Zahl $\lambda \in \mathbb{N}$ und ein ungerichteter λ-fach kantenzusammenhängender Graph G. Wir suchen einen minmalen λ-fach kantenzusammenhängenden Teilgraphen von G und betrachten zunächst den Fall, dass G gewichtet ist. Hinterher gehen wir auf den Spezialfall ein, wenn G ein ungewichteter Graph ist.

2.1 Der gewichtete Fall

Wir präsentieren einen Algorithmus aus [7]. Die Idee ist es, das Problem auf folgendes Optimierungsproblem zurückzuführen: Gegeben ein gewichteter, gerichteter Graph G^D, eine Wurzel $r \in V(G^D)$, sowie eine Zahl $\lambda \in \mathbb{N}$, finde einen minimalen Teilgraphen $H^D \subseteq G^D$, der für alle $v \in V(G^D)$ mindestens λ kantenunabhängige Pfade von r zu v hat. Dieses Problem lässt sich exakt in Polynomzeit lösen (siehe [4], [2]). Der Algorithmus für das ursprüngliche Problem geht dann wiefolgt vor:

WEIGHTED EDGE-CONNECTIVITY

(1) Wandle G in einen gerichteten Graphen G^D um, indem jede Kante $\{u, v\} \in E(G)$ durch zwei Kanten $(u, v), (v, u)$ jeweils mit Gewicht $\omega(\{u, v\})$ ersetzt wird, und wähle eine beliebige Wurzel $r \in V(G^D)$.

(2) Wende oben beschriebenen Algorithmus auf G^D an und erhalte $H^D \subseteq G^D$.

(3) Wandle H^D in einen ungerichteten Graphen H um, wobei die Kanten von H durch $E(H) = \big\{\{u, v\} \mid (u, v) \in E(H^D) \text{ oder } (v, u) \in E(H^D)\big\}$ definiert sind.

(4) Gib H zurück.

Wir beweisen die Korrektheit des Algorithmus und die Approximationsgüte von 2 mithilfe der folgenden 2 Lemmas:

Lemma 2. *H ist λ-fach kantenzusammenhängend.*

Beweis. Angenommen, H sei nicht λ-fach kantenzusammenhängend. Dann gibt es einen Schnitt $C \subseteq E(H)$ von höchstens $\lambda - 1$ Kanten (d.h. $|C| \leq \lambda - 1$), der H in mindestens zwei verschiedene Zusammenhangskomponenten $A, B \subseteq V(H)$ trennt. O.B.d.A. sei $r \in A$. Da B nichtleer ist, existiert ein $v \in B$. Da in H^D mindestens λ kantenunabhängige (gerichtete) Pfade von r nach v existieren, muss es auch in H mindestens λ solche kantenunabhängigen Pfade geben. Dies ist ein Widerspruch dazu, dass C ein Schnitt ist, da mindestens einer der Pfade nach dem Entfernen von C noch vollständig im Graphen H vorhanden sein muss.

Lemma 3. *Sei OPT das Gewicht einer optimalen Lösung. Dann gilt $\omega(E(H)) \leq 2OPT$.*

Beweis. Sei $G_O \subseteq G$ eine optimale Lösung (d.h. $\omega(E(G_O)) = OPT$). Wandle nun G_O wie in Schritt 1 des Algorithmus in einen gerichteten Graphen G_O^D um. Dann ist $G_O^D \subseteq G^D$ und $\omega(E(G_O^D)) = 2OPT$. Da der ungerichtete Graph G_O λ-fach kantenzusammenhängend ist, gibt es in dem gerichteten Graphen G_O^D für alle Knoten $r \in V(G_O^D)$ mindestens λ kantenunabhängige gerichtete Pfade zu jedem anderen Knoten $v \in V(G_O^D)$. Der in WEIGHTED EDGE-CONNECTIVITY in Schritt (3) gefundene Teilgraph H^D hat daher höchstens ein genauso großes Gewicht wie G_O^D, d.h. $\omega(E(H^D)) \leq \omega(E(G_O^D)) = 2OPT$. Nach Konstruktion gilt zudem $\omega(E(H)) \leq \omega(E(H^D)$, woraus sofort die Behauptung folgt.

2.2 Der ungewichtete Fall

Wir betrachten nun den Fall, bei dem der Eingabegraph G ungewichtet ist. Man beachte, dass wir den Algorithmus aus dem vorigen Abschnitt für diesen Fall übernehmen können und er auch hier eine Appro-

ximationsgüte von 2 liefert. Wir können diese Güte mit etwas Arbeit jedoch noch etwas verbessern. Um genauer zu sein werden, wir einen Algorithmus mit Approximationsgüte $2 - 1/\lambda$ vorstellen.
In [8] wird ein Algorithmus beschrieben, der für einen ungewichteten, ungerichteten Graphen G einen λ-fach kantenzusammenhängenden Teilgraphen mit höchstens $\lambda|V(G)|$ Kanten ausgibt. Dieser liefert uns für den ungewichteten Fall neben dem Algorithmus aus dem vorigen Abschnitt einen weiteren 2-Approximationsalgorithmus, denn: In einer optimalen Lösung G_O muss jeder Knoten mindestens Grad λ haben, also $E(G_O) \geq \frac{|V(G)|\lambda}{2}$.
Die Idee, um diese Approximationsgüte auf $2 - 1/\lambda$ zu drücken, ist wiefolgt: Wir möchten einen "guten Starter"finden, d.h. einen "guten"2-fach zusammenhängenden Teilgraphen, den wir dann zu einem λ-fach kantenzusammenhängenden Teilgraphen erweitern. Der Gewinn von $1/\lambda$ in der Approximationsgüte wird dabei genau durch den "guten Starterërreicht. Damit ist auch klar, warum der Gewinn für große λ immer unbedeutender wird.

2.2.1 Ein guter Starter: Erweiterte Tiefensuche

Den guten Start erreichen wir, indem wir eine modifizierte Tiefensuche (DFS, **D**epth **F**irst **S**earch) auf dem Graphen ausführen. Für eine formale Beschreibung des Algorithmus verweisen wir auf [7], wir wollen hier eine ausführliche, unformale Beschreibung angeben.
Die Idee ist, dass uns die Tiefensuche einen 1-fach kantenzusammenhängenden Teilgraphen (nämlich einen Baum) liefert und wir daraus einen 2-fach kantenzusammenhängenden Graphen machen können, indem wir jede Baumkante durch eine weitere "absichern". Dieses Absichern geschieht in dem Moment, in dem wir einen Knoten v bei der Tiefensuche endgültig verlassen wollen. Wir überprüfen dann, ob die Kante e von v zu seinem Vorgänger u im aktuellen Teilgraphen eine Brücke – d.h. eine Kante, deren Entfernen den Graphen trennt – ist. Ist dies der Fall, so muss diese Kante e noch abgesichert werden und wir fügen zum aktuellen Teilgraphen die "beste" Kante hinzu, die e absichert. Was ist nun die "beste" Kante? Nun, da wir eine Tiefensuche durchführen, wissen wir, dass wir alle Knoten aus dem Unterbaum von v schon endgültig verlassen haben und deren Kanten schon abgesichert haben (d.h. der Unterbaum von v ist bereits 2-fach kantenzusammenhängend). Es gibt aber noch Kanten zwischen der Baumwurzel und v, die potenzielle Brücken sind. Die "beste" Kante ist in diesem Moment also diejenige, die am meisten von den weiteren potenziellen Brücken mit absichert. Konkret heißt das, dass wir die Kante suchen, die von v oder aus dem Unterbaum von v so nah an die Wurzel herankommt wie möglich. Alle Kanten bis dorthin sind dann offensichtlich auch abgesichert und somit keine Brücken mehr.
Es sei angemerkt, dass man mit entsprechenden Datenstrukturen diesen Algorithmus in linearer Laufzeit implementieren kann (siehe [7]). Wir nennen diesen Algorithmus X-DFS. Es ist klar, dass dieser einen 2-fach kantenzusammenhängenden Graphen liefert. Zudem gilt:

Lemma 4. *Sei OPT die minimale Anzahl Kanten eines 2-fach kantenzusammenhängenden Teilgraphen von G. Dann gilt $|E(\text{X-DFS}(G))| \leq 3/2OPT$.*

Beweis. Um $\frac{|E(\text{X-DFS}(G))|}{OPT}$ nach oben abzuschätzen, benötigen wir untere Schranken für OPT. Offensichtlich gilt $OPT \geq n$, da der zu OPT gehörende Graph 2-fach kantenzusammenhängend sein muss. Wir wollen zusätzlich $OPT \geq 2k$ zeigen, wobei k die Anzahl der Rückkanten ist, die wir in X-DFS zur Absicherung hinzufügen. Dazu definieren wir folgende Partition des Graphen G: Bezeichne mit T den Baum, der bei der Tiefensuche von X-DFS entsteht (also den Ausgabegraphen ohne die zur Absicherung hinzugefügten Rückkanten). Entferne aus T die Kanten, die in der Ausführung von X-DFS als Brücken erkannt wurden und abgesichert werden mussten. Dieses Entfernen spaltet T in genau $k + 1$ Zusammenhangskomponenten $C_1, C_2, \ldots, C_{k+1}$ auf. Diese bilden eine Partition von $V(G)$.
Man kann sich nun folgendes überlegen: Da wir in X-DFS stets die Kante zur Absicherung genommen haben, die uns so nah wie möglich zur Wurzel bringt, ist der "von G induzierte" Graph auf diesen Komponenten – formal definiert durch $G' = (V', E')$ mit $V' = \{C_1, \ldots, C_{k+1}\}, E' = \{\{C_i, C_j\} \mid \exists u \in C_i, v \in C_j : \{u, v\} \in E(G)\}$ – ein Baum. Das heisst aber, dass jeder 2-fach kantenzusammenhängende Teilgraph von G zwischen zwei benachbarten Komponenten C_i und C_j mindestens zwei Kanten beinhalten muss! Andernfalls könnte man durch das Entfernen einer Kante den Graphen trennen. Wegen $|E'| = 2k$ gilt damit $OPT \geq 2k$.

Da nun der Ausgabegraph von X-DFS genau $n-1+k$ Kanten enthält ($n-1$ vom Tiefensuchebaum und k Kanten zur Absicherung), erhalten wir die Behauptung durch eine einfache Fallunterscheidung:

$$\frac{n-1+k}{OPT} \leq \frac{n-1+k}{\max\{n, 2k\}} \leq \begin{cases} \frac{2k-1+k}{2k} < \frac{3}{2} & \text{falls } 2k \geq n \\ \frac{n-1+n/2}{n} < \frac{3}{2} & \text{falls } n/2 > k. \end{cases}$$

2.2.2 Der vollständige Algorithmus

Um den vollständigen Algorithmus aus [8] angeben zu können, benötigen wir noch etwas Notation:

Definition 5. *Ein* Wald *ist eine Menge von Bäumen. In einem Graphen G heißt ein Wald $W \subseteq G$* maximal spannend, *falls jeder Wald $W' \subseteq G$ mindestens genau so viele Zusammenhangskomponenten wie W enthält.*

Ein maximal spannender Wald enthält also so viele Kanten wie möglich und minimert damit die Anzahl der Zusammenhangskomponenten.

Definition 6. *Sei G ein Graph und $H \subseteq G$. Mit $G - H$ oder $G - E(H)$ bezeichnen wir den Graphen $G' = (V', E')$ mit $V' = V(G)$ und $E' = E(G) \setminus E(H)$.*

UNWEIGHTED EDGE-CONNECTIVITY

Eingabe: $\lambda \in \mathbb{N}_{\geq 2}$ und ein ungerichteter, ungewichteter λ-fach kantenzusammenhängender Graph G

(1) Setze $H_2 = $ X-DFS(G).

(2) **for** $i = 3, \ldots, \lambda$
  Finde einen maximal spannenden Wald W_i in $G - H_{i-1}$
  Setze $H_i = H_{i-1} \cup W_i$.

(3) Gib H_λ zurück.

Erneut zeigen wirdie Korrektheit und Approximationsgüte durch zwei Lemmas.

Lemma 7. *Der Ausgabegraph H_λ ist λ-fach kantenzusammenhängend.*

Beweis. Angenommen, dies wäre nicht der Fall, d.h. es existiert ein Schnitt $S \subseteq E(H_\lambda)$ mit $|S| < \lambda$, so dass $H_\lambda - S$ mindestens zwei Zusammenhangskomponenten H_1, H_2 mit $H_1 \neq H_2$ hat. In Schritt 1 werden genau zwei Kanten zwischen H_1 und H_2 zu H_λ hinzugefügt. Diese Kanten müssen beide in S enthalten sein, da H_λ durch S getrennt wird. Somit muss es aber einen Wald W_i geben, der keine Kante aus S enthält (d.h. $E(W_i) \cap S = \emptyset$). Da der Eingabegraph G jedoch λ-fach kantenzusammenhängend ist, existiert in $G - S$ noch mindestens eine Kante zwischen H_1 und H_2. Eine solche Kante $e \in E(G) \setminus S$ ist insbesondere auch in $E(W_i)$ enthalten, da W_i maximal spannend ist. Also ist auch $e \in E(H_\lambda)$ und somit H_1 mit H_2 verbunden, also $H_1 = H_2$. (Widerspruch)

Lemma 8. UNWEIGHTED EDGE-CONNECTIVITY *hat eine Approximationsgüte von $2 - 1/\lambda$.*

Beweis. Wir bezeichnen mit k wieder die Anzahl der in Schritt 1 hinzugefügten Rückkanten und mit OPT die Anzahl der Kanten in einer optimalen Lösung. Mit dem Argument aus dem Beweis von Lemma 5 erhalten wir $OPT \geq \lambda k$. Da in einem λ-fach kantenzusammenhängenden Graphen jeder Knoten mindestens Grad λ haben muss, erhalten wir zudem $OPT \geq \frac{\lambda n}{2}$. Zudem ist die Anzahl der Kanten in H_λ durch $n - 1 + k + (\lambda - 2)(n - 1)$ beschränkt ($n - 1 + k$ Kanten aus Schritt 1 und höchstens $n - 1$ weitere aus jeder Iteration). Somit gilt

$$\frac{n-1+k+(\lambda-2)(n-1)}{OPT} \leq \frac{n-1+k+\lambda n-2n-\lambda+2}{\max\{(\lambda n)/2, \lambda k\}} = \frac{\lambda n-(n-k+\lambda-1)}{\max\{(\lambda n)/2, \lambda k\}}$$
$$\overset{\lambda-1\geq 0}{\leq} \frac{\lambda n}{(\lambda n)/2} - \frac{n-k}{\max\{(\lambda n)/2, \lambda k\}} \leq \begin{cases} 2 - \frac{k}{\lambda k} \leq 2 - \frac{1}{\lambda} & \text{falls } 2k \leq n \\ 2 - \frac{n/2}{(\lambda n)/2} \leq 2 - \frac{1}{\lambda} & \text{falls } n/2 < k \end{cases}$$

Anmerkung. Mit einer geschickten Implementierung kann Schritt (2) in einer Iteration und somit in Linearzeit durchgeführt werden. Damit hat auch der gesamte Algorithmus lineare Laufzeit.

2.3 Das Knotenzusammenhangsproblem

Zum Problem, einen minimalen λ-fach *knoten*zusammenhängenden Teilgraphen zu finden, sei angemerkt, dass es im gewichteten Fall noch keinen Algorithmus mit konstanter Approximationsgüte gibt. Der beste bekannte Algorithmus ist aus [10] und hat eine Approximationsgüte von $2H(\lambda)$, wobei $H(\lambda) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{\lambda}$ die harmonische Zahl von λ ist.
Der in Abschnitt 2.1 beschriebene Algorithmus kann für $\lambda = 2$ leicht modifiziert werden und liefert in diesem Spezialfall dann einen 2-Approximationsalgorithmus für das Knotenzusammenhangsproblem. Für Details verweisen wir auf [3] und [9].
Im ungewichteten Fall gibt es einen ähnlichen Algorithmus zu dem aus Abschnitt 2.2. Der X-DFS Algorithmus muss etwas modifiziert werden, damit der Ausgabegraph auch 2-fach knotenzusammenhängend ist (siehe dazu [7]). Zudem ist in Schritt 2 die Auswahl der maximal spannenden Bäume nicht mehr beliebig. Nur bestimmte Wälder (z.B. Breitensuche-Wälder) garantieren den λ-fachen Knotenzusammenhang des Ausgabegraphen. Für genauere Ausführungen sei auf [1] verwiesen.

3 Das Problem des starken Zusammenhangs

Wir widmen uns nun dem Problem einen minimalen stark zusammenhängenden Teilgraphen zu finden. Als Eingabe erhalten wir dabei einen gerichteten Graphen, der stark zusammenhängend ist. Wir beschreiben einen Approximationsalgorithmus, der auf [6] zurückgeht. Die Idee des Algorithmus ist so lange längste Kreise im Graphen zusammenzuziehen, bis nur noch ein Knoten übrig bleibt. In diesem Abschnitt sei G stets ein gerichteter, stark zusammenhängender Graph.

Definition 9. *Eine Kante $(u, v) \in E(G)$ zusammenzuziehen bedeutet die Knoten u und v durch einen einzelnen Knoten zu ersetzen und evtl. entstandene Schleifen oder Mehrfachkanten durch einfache Kanten zu ersetzen. Ein Menge von Kanten zusammenzuziehen bedeutet die Kanten in einer beliebigen Reihenfolge zusammenzuziehen.*

CONTRACT CYCLES$_k$

Eingabe: ein gerichteter, ungewichteter stark zusammenhängender Graph G

(1) **for** $i = k, k-1, \ldots, 2$
  while (Der Graph enthält einen Kreis der Länge $\geq k$)
    Ziehe die Kanten eines solchen Kreises zusammen.

(3) Bezeichne die Menge der zusammengezogenen Kanten mit E' und gibt $G' = (V, E')$ zurück.

Es ist offentsichtlich, dass CONTRACT CYCLES$_k$ einen stark zusammenhängenden Graphen zurückgibt. Damit bleibt noch die Approximationsgüte zu zeigen. Zunächst geben wir eine Abschätzung einer optimalen Lösung an. Diese besagt, dass ein Graph, der lediglich kurze Kreise enthält, viele Kanten braucht, um stark zusammenhängend zu sein.

Lemma 10. *Sei $n = |V(G)|$ und sei c die Länge des längsten Kreises in G. Sei zudem OPT die Anzahl Kanten eines minimalen stark zusammenhängenden Teilgraphen von G. Dann gilt*

$$OPT \geq \frac{c}{c-1}(n-1).$$

Beweis. Sei E_O die Kantenmenge einer optimalen Lösung. Ziehe im Graphen $(V(G), E_O)$ nach und nach Kreise $C_1, C_2, \ldots C_m$ zusammen, bis nur noch ein einelner Knoten übrig bleibt. Es gilt

$$|E_O| = (m-1)\frac{|E_O|}{n-1} = (n-1)\frac{\sum_{i=1}^{n}|C_i|}{\sum_{i=1}^{n}(|C_i|-1)} \geq (n-1)\frac{c}{c-1}.$$

Damit können wir nun die Güte unseres Algorithmus abschätzen.

Lemma 11. *Sei $k \in \mathbb{N}_{\geq 2}$ und sei OPT die Anzahl Kanten in einem minimalen stark zusammenhängenden Teilgraphen von G. Dann gilt*

$$|E(\textsc{Contract Cycles}_k(G))| \leq c_k \cdot OPT \qquad \text{, wobei } \frac{\pi^2}{6} \leq c_k \leq \frac{pi^2}{6} + \frac{1}{(k-1)k}.$$

Beweis. Wir setzen zunächst $n := |V(G)|$. Mit G_i bezeichnen wir den Graphen in der Ausführung des Algorithmus unmittelbar nach dem Zusammenziehen aller Kreise der Länge $\geq i$ und definieren $n_i := |V(G_i)|$. Im Beweis zu Lemma 11 konnten wir die Anzahl der Kanten nach unten abschätzen, da Kreise der Länge $\leq c$ zusammengezogen wurden. Mit dem selben Argument können wir die Anzahl Kanten auch nach oben abschätzen, wenn wir Kreise der Länge $\geq c$ zusammenziehen. Damit ergibt sich:

$$|E(\textsc{Contract Cycles}_k(G))| \leq \frac{k}{k-1}(n - n_k) + \sum_{i=2}^{k-1} \frac{i}{i-1}(n_{i+1} - n_i)$$

Sei nun $OPT(G_i)$ die minimale Anzahl an Kanten in einem stark zusammenhängenden Teilgraphen von G_i. Dann gilt nach Lemma 11

$$OPT \geq OPT(G_i) \geq \frac{i-1}{i-2}(n_i - 1).$$

Trivialerweise gilt auch $OPT \geq n$. Eine längere Rechnung mit elementaren Schritten und dem aus der Analysis bekannten Ergebnis $\sum_{i=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$ liefert die Behauptung.
Wir erhalten damit für $\textsc{Contract Cycles}_k$ folgende Approximationsgüten:

$k = 3 : 1.750$ $\quad$ $k = 4 : 1.694$ $\quad$ $k = 5 : 1.674$ $\quad$ $k = \infty : 1.645$

Es sei angemerkt, dass in [6] auch darauf eingegangen wird, wie $\textsc{Contract Cylces}_3$ mit geeigneten Datenstrukturen (genauer: union-find Datenstruktur für die Verwaltung der zusammengezogenen Kreise) in quasi linearer Laufzeit implementiert werden kann.

Literatur

[1] J. Cheriyan and R. Thurimella. Algorithms for parallel k-vertex connectivity and sparse certificates. *Proc. 23rd Annual Symposium on Theory of Computing*, pages 391–401, 1991.

[2] J. Edmonds. Matroid intersection. In *Annals of Discrete Mathematics*, volume 4, pages 39–49, 1979.

[3] A. Frank and E. Tardos. An application of submodular flows. *Linear Algebra and its Applications*, 114/115:320–348, 1991.

[4] H. N. Gabow. A matroid approach to finding edge connectivity and packing arborescences. *Proc. 23rd Annual Symposium on Theory of Computing*, pages 112–122, 1991.

[5] Dorit S. Hochbaum (Hrsg.). *Approximation algorithms for $\mathcal{NP}$-hard problems*. PWS Publishing Company, 1995.

[6] S. Khuller, B. Raghavachari, and N. Young. Approximating the minimum equivalent digraph. *SIAM Journal on Computing*, 24(4):859–872, 1995.

[7] S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of Algorithms*, 14(2):214–225, 1993.

[8] H. Nagamochi and T. Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(5/6):583–596, 1992.

[9] M. Penn and H. Shasha-Krupnik. Improved approximation algorithms for weighted 2 & 3 vertex connectivity augmentation problems. Technical Report TR-95-IEM/OR-1, Industrial Engineering and Management, Technion, Haifa, Israel, May 1995.

[10] R.Ravi and D. Williamson. An approximation algorithm for minimum-cost vertex-connectivity problems. *Proc. 6th Annaul ACM-SIAM Symposium on Discrete Algorithms,* pages 332–341, 1995.

Anhang

A Elementare Definitionen und Sätze zu Graphen

Definition A.1. *Ein gerichteter Graph $G = (V, E)$ besteht aus einer enldichen Menge V von Knoten und einer Menge $E \subseteq V \times V$ von Kanten (manchaml auch $V(G)$ und $E(G)$). Im Falle eines ungerichteten Graphen gilt $E \subseteq \binom{V}{2} = \{\{a, b\} \mid a, b \in V$ und $a \neq b\}$.*

Von nun an sei G ein Graph.

Definition A.2. *Ein gewichteter Graph G verfügt zusätzlich über eine Gewichtungsfunktion $\omega : E(G) \to \mathbb{R}_{\geq 0}$, die jeder Kante ein nicht-negatives Gewicht zuordnet.*

Definition A.3. *Ein gerichteter Graph G heißt stark zusammenhängend, falls für alle $u, v \in V(G)$ ein gerichteter Pfad von u nach v existiert.*

Definition A.4. *Ein* Pfad *in G ist eine Liste $(v_1, v_2, \ldots, v_n)$, wobei neben $n \in \mathbb{N}$ gelten muss:*

- *$v_i \in V(G)$ für alle $i \in \{1, \ldots, n\}$,*
- *$(v_i, v_{i+1}) \in E(G)$ für alle $i \in \{1, \ldots, n-1\}$ in gerichteten Graphen,*
- *$\{v_i, v_{i+1}\} \in E(G)$ für alle $i \in \{1, \ldots, n-1\}$ in ungerichteten Graphen,*
- *Alle Knoten außer v_1 und v_n müssen verschieden sein, d.h. $v_i \neq v_j$ für alle $\{1, n\} \neq \{i, j\} \subseteq \{1, \ldots, n\}$.*

Definition A.5. *Ein* Kreis *ist ein Pfad $(v_1, v_2, \ldots, v_n)$ mit $v_1 = v_n$.*

Definition A.6. *Ein* Schnitt *$C \subseteq E(G)$ oder $C \subseteq V(G)$ ist eine Menge von Kanten oder Knoten, nach deren Entfernen aus dem Graphen der Graph mehrere verschiedene Zusammenhangskomponenten besitzt.*

Definition A.7. *G heißt genau dann Baum, wenn G zusammenhängend ist und keine Kreise enthält.*

Satz A.8. *Für jeden Baum G gilt: $|E(G)| = |V(G)| - 1$.*

Definition A.9. *Eine* Zusammenhangskomponente *$Z \subseteq V(G)$ ist eine maximale zusammenhängende Teilmenge der Knoten von G.*

Definition A.10. *Eine* Brücke *ist eine Kante $e \in E(G)$, so dass $C := \{e\}$ ein Schnitt ist.*

Lemma A.11. *Sei G ein Baum und $e \in E(G)$. Das Entfernen von e trennt den Baum in zwei Zusammenhangskomponenten.*