ETH zürich
Departement Informatik

Algorithmik für schwere Probleme

Prof. Dr. Dennis Komm
Dr. Hans-Joachim Böckenhauer
`https://courses.ite.inf.ethz.ch/schwere_prob_23/`

Lösungsvorschläge – Blatt 5

Zürich, 28. März 2023

Lösung zu Aufgabe 5

(a) Wenn wir im ersten Schritt die Datenreduktionsregel anwenden, in der alle Knoten mit Grad grösser als k entfernt werden, ändert sich dadurch der Graph nicht, da alle Knoten höchstens Grad $k-1$ haben. Die Laufzeit dieses Schrittes ist mit einer geeigneten Datenstruktur in $\Theta(n) = \Theta(k^2)$.

Wenn wir dann im zweiten Schritt den Divide-and-Conquer-Algorithmus anwenden und dieser die Kanten des langen Pfades von rechts nach links durchgeht ($\{u_{3k+1}, u_{3k}\}$, $\{u_{3k}, u_{3k-1}\}$, usw.), hat der Suchbaum maximale Grösse, d. h. er enthält $2^{k+1} - 2$ Kanten. Bei jeder Kante entfernen wir einen Knoten samt aller adjazenten Kanten aus dem Graphen; dies ist in Zeit $\Theta(n) = \Theta(k^2)$ möglich.

Insgesamt erhalten wir also eine Laufzeit von $\Theta(k^2 + (2^k - 2) \cdot k^2) = \Theta(2^k k^2)$.

(b) Eine bedeutend bessere Laufzeit erhalten wir, wenn wir die Datenreduktionsregel nach jedem Schritt des Divide-and-Conquer-Algorithmus anwenden. Da der Parameter mit jedem Schritt verringert wird, entfernt die Datenreduktionsregel spätestens im dritten Schritt den gesamten „Kopf“

$$G[\{r\} \cup \{v_i \mid 1 \leq i \leq k-2\} \cup \{w_{i,j} \mid 1 \leq i,j \leq k-2\} \cup \{u_1\}]$$

und gibt einen Problemkern $(G, 1)$ für einen induzierten Teilgraphen G von G_k aus. Das Entfernen des Kopfes ist in linearer Zeit möglich (bezüglich $|V| = k^2 + 4$), und der Kern $(G, 1)$ kann in konstanter Zeit entschieden werden.

Da die Grösse des Suchbaums bei diesem Ansatz konstant ist, ist die Laufzeit des Algorithmus somit in $\Theta(k^2)$.

Lösung zu Aufgabe 6

(a) Regel R1 ist sicher, weil ein Knoten v mit einer Schleife in jedem Feedback-Vertex-Set enthalten sein muss. Wenn wir also ein Feedback-Vertex-Set der Grösse $k-1$ für den Multigraphen $G - v$ finden, dann erhalten wir daraus ein Feedback-Vertex-Set der Grösse k für G, indem wir v hinzufügen.

Regel R2 ist sicher, weil man alle Kreise, die durch eine Mehrfachkante entstehen, gleichzeitig aufbrechen kann, indem man einen der beiden Endpunkte der Mehrfachkante in das Feedback-Vertex-Set aufnimmt. Deshalb ändert sich die Menge der zulässigen Feedback-Vertex-Sets nicht, wenn man jede Mehrfachkante durch eine Doppelkante ersetzt.

Regel R3 ist sicher, weil ein Knoten v vom Grad 1 nicht Teil eines Kreises sein kann und das Entfernen von v damit die Kreisstruktur des Graphen nicht ändert. Jedes Feedback-Vertex-Set, das v enthält, bleibt auch ohne v ein Feedback-Vertex-Set.

Für Regel R4 bemerken wir zunächst, dass ein Knoten v, auf den diese Regel angewendet wird, keine Schleife haben kann, da sonst zuerst Regel R1 angewendet würde. Falls v der Endpunkt einer Doppelkante ist, verlangt die Regel R4, dass wir ihn löschen und an dem anderen Endpunkt u dieser Doppelkante eine Schleife hinzufügen (die dann anschliessend die Anwendung von Regel R1 auslöst). Falls v in einem Feedback-Vertex-Set enthalten ist, dann verhindert es dort genau den einen Kreis, der durch die Doppelkante gebildet wird. Derselbe Effekt lässt sich erreichen, wenn man in dem Feedback-Vertex-Set den Knoten v durch u ersetzt. Wenn die beiden Nachbarn von v verschieden sind, ist es leicht zu sehen, dass die Transformation gemäss Regel R4 die Kreisstruktur des Graphen nicht verändert. Somit ist auch diese Regel sicher.

(b) Wenn X ein Feedback-Vertex-Set für G ist, dann ist $H = G - X$ nach Definition ein Wald. Damit ist die Anzahl der Kanten in H höchstens $E(H) \leq |V| - |X| - 1$. Jede Kante in $E - E(H)$ ist inzident zu einem Knoten aus X. Damit folgt

$$\sum_{v \in X} \deg(v) + |V| - |X| - 1 \geq |E|$$

und somit

$$D(X) = \sum_{v \in X} (\deg(v) - 1) \geq |E| - |V| + 1 \; .$$

(c) Für einen Widerspruchsbeweis nehmen wir an, dass es ein Feedback-Vertex-Set X der Grösse k in G gibt mit $X \cap V_{3k} = \emptyset$. Aufgrund der Definition von V_{3k} hat jeder Knoten $v \notin V_{3k}$ einen Grad von $\deg(v) \leq \min_{w \in V_{3k}}\{\deg(w)\}$. Weil ausserdem $|X| \leq k$ gilt, können wir die Aussage aus Aufgabenteil (b) anwenden und erhalten

$$D(V_{3k}) \geq 3k \cdot (\deg(v) - 1) \geq 3 \cdot D(X) \geq 3 \cdot (|E| - |V| + 1) \; .$$

Weiterhin gilt $X \subseteq V \setminus V_{3k}$ und deshalb

$$D(V \setminus V_{3k}) \geq D(X) \geq |E| - |V| + 1 \; .$$

Insgesamt erhalten wir

$$D(V) = D(V_{3k}) + D(V - V_{3k}) \geq 4 \cdot (|E| - |V| + 1) \; .$$

Weil aber $\sum_{i=1}^{n} \deg(v_i) = 2|E|$ gilt, da in dieser Summe jede Kante genau zweimal gezählt wird, folgt

$$D(V) = 2|E| - |V| \geq 4 \cdot (|E| - |V| + 1) \; .$$

Hieraus folgt unmittelbar, dass $0 \geq 2|E| - 3|V| + 4$, also $3|V| > 2|E|$. Dies ist ein Widerspruch dazu, dass nach Anwendung der Reduktionsregeln der Grad eines jeden Knotens mindestens 3 ist, denn dann würde es mindestens $(3/2)|V|$ Kanten geben.

(d) Wir betrachten den folgenden Algorithmus: Auf eine Eingabe (G, k) werden zunächst die Reduktionsregeln R1 bis R4 solange wie möglich angewendet. Falls danach $k' < 0$ gilt, bricht der Algorithmus mit der Ausgabe NEIN ab. Falls nicht, dann haben wir eine äquivalente Instanz (G', k') erhalten, so dass G' einen minimalen Knotengrad von 3 hat und $k' \leq k$ gilt. Falls G' keine Kanten enthält, gibt der Algorithmus JA aus, weil ein Feedback-Vertex-Set der Grösse $0 \leq k'$ ausreicht, um alle Kreise zu überdecken.

Falls die Anzahl Knoten von G' mindestens $3k'$ ist, sei $V_{3k'}$ wie in Aufgabenteil (c) definiert. Wie wir dort gesehen haben, enthält jedes Feedback-Vertex-Set mindestens einen Knoten aus $V_{3k'}$. Der Algorithmus verzweigt nun über die Auswahl dieses Knotens und löst rekursiv FEEDBACKVERTEXSET auf den Instanzen $(G' - v, k' - 1)$ für alle $v \in V_{3k'}$ oder für alle $v \in V$ falls $n < 3k'$. Falls einer dieser rekursiven Aufrufe eine Lösung findet, dann ist auch die ursprüngliche Instanz lösbar und der Algorithmus gibt JA zurück. Wenn alle Aufrufe NEIN zurückgeben, gibt auch der Algorithmus NEIN zurück.

Aus den Aufgabenteilen (a) und (c) folgt unmittelbar die Korrektheit des Algorithmus.

Für die Laufzeitanalyse bemerken wir, dass der Parameter k bei jedem rekursiven Aufruf um mindestens 1 verringert wird. Also ist die Höhe des Suchbaums durch k beschränkt. In jedem Schritt der Rekursion verzweigt der Algorithmus in höchstens $3k$ Teilprobleme, also hat der Suchbaum höchstens $(3k)^k$ Knoten. In jedem der Knoten muss der Algorithmus nur die Reduktionsregeln anwenden, was in polynomieller Zeit möglich ist.[1] Damit ist die Laufzeit insgesamt nicht grösser als $(3k)^k \cdot p(n)$ für ein geeignetes Polynom p.

[1] Wenn die Mehrfachheit einer Kante nicht polynomiell in n ist, können wir die Zahl unter Umständen nicht einmal in Polynomzeit lesen. Wir setzen daher voraus, dass dies nicht der Fall ist.