

Lösungsvorschläge – Blatt 10

Zürich, 6. Mai 2025

Lösung zu Aufgabe 11

- (a) In der Vorlesung wurde gezeigt, dass eine Baumzerlegung $(T, \{X_i \mid i \in I\})$ des Graphen G genau dann einfach ist, wenn für jede Kante $\{X_i, X_j\}$ in T gilt, dass $X_i \not\subseteq X_j$ und $X_j \not\subseteq X_i$. Also genügt es, alle Kanten $\{X_i, X_j\}$ einer Baumzerlegung zu inspizieren und diejenigen, welche die Bedingung für eine einfache Baumzerlegung nicht erfüllen, zu kontrahieren, d. h. durch eine neue Bag X_{ij} zu ersetzen mit $X_{ij} = X_i \cup X_j$. Aus $X_i \subseteq X_j$ oder $X_j \subseteq X_i$ folgt, dass $|X_{ij}| = \max\{|X_i|, |X_j|\}$ gilt, weshalb die Weite der einfachen Baumzerlegung erhalten bleibt.

Wir zeigen nun, dass es genügt, jede Kante e nur einmal zu inspizieren. Wenn also eine Kante $e = \{X_i, X_j\}$ die Bedingung $X_i \not\subseteq X_j$ und $X_j \not\subseteq X_i$ erfüllt, dann erfüllt sie die Bedingung auch, nachdem andere Kanten mit Endpunkten X_i bzw. X_j kontrahiert wurden. Wir nehmen o.B.d.A. an, dass eine Kante $e' = \{X_i, X_k\}$ kontrahiert wurde, wodurch nun $X_i \cup X_k \subseteq X_j$ oder $X_j \subseteq X_i \cup X_k$ gilt. Aus $X_i \cup X_k \subseteq X_j$ folgt insbesondere $X_i \subseteq X_j$, was sofort einen Widerspruch zu der Bedingung $X_i \not\subseteq X_j$ für die Kante e ergibt. Es muss also $X_j \subseteq X_i \cup X_k$ gelten. Da die Kante $\{X_i, X_k\}$ kontrahiert wurde, gilt $X_i \subseteq X_k$ oder $X_k \subseteq X_i$. Falls $X_k \subseteq X_i$, ergibt sich aus $X_j \subseteq X_i \cup X_k$ einen Widerspruch zu der Bedingung $X_j \not\subseteq X_i$ für die Kante e . Es gilt also $X_i \subseteq X_k$. Aus $X_j \subseteq X_i \cup X_k$ folgt dann $X_j \subseteq X_k$. Wir betrachten schliesslich den Pfad X_j, X_i, X_k , der aus den Kanten $e = \{X_j, X_i\}$ und $e' = \{X_i, X_k\}$ besteht. Aus $X_j \subseteq X_k$ und der Eigenschaft (iii) einer Baumzerlegung folgt, dass $X_j \subseteq X_i$, was einen Widerspruch zu der Bedingung $X_j \not\subseteq X_i$ für die Kante e ergibt.

Eine Kante kann in Zeit $\mathcal{O}(k)$ überprüft und bei Bedarf kontrahiert werden. Insgesamt gibt es im Baum T genau $|E(T)| = |V(T)| - 1$ Kanten. Eine einfache Baumzerlegung kann somit in Zeit $\mathcal{O}(k \cdot |V(T)|)$ berechnet werden.

- (b) Die Baumzerlegung hat Weite k , also hat jede Bag höchstens Grösse $k + 1$. Ein Algorithmus muss jede mögliche 3-Färbung dieser Menge betrachten und überprüfen, ob sie gültig ist. Es gibt für eine Knotenmenge der Kardinalität höchstens $k + 1$ höchstens 3^{k+1} viele verschiedene 3-Färbungen. Die Überprüfung, ob eine 3-Färbung gültig ist, kann mit einer geeigneten Datenstruktur (z. B. einer Adjazenzliste) in Zeit $\mathcal{O}(k^2)$ erfolgen. Das lässt sich aber noch verbessern, indem die Färbungen

in einer geeigneten Reihenfolge durchlaufen werden. Wenn sich zum Beispiel von einer Färbung zur nächsten jeweils nur die Farbe eines Knoten verändert (analog zur Verwendung des Gray-Codes im Algorithmus für Vertex Cover), muss nur die Nachbarschaft dieses Knoten neu überprüft werden. Die Gesamtlaufzeit des Algorithmus pro Bag ist somit in $\mathcal{O}(3^k \cdot k)$.

(c) Für jedes Blatt X_i in T gilt offensichtlich $\text{Extcol}(X_i) = \text{Col}(X_i)$.

Für einen beliebigen inneren Knoten X nehmen wir an, $\text{Extcol}(X_1), \dots, \text{Extcol}(X_m)$ wurden bereits berechnet, wobei X_1, \dots, X_m die Kinder von X sind. Dann besteht $\text{Extcol}(X)$ aus allen Färbungen $c \in \text{Col}(X)$, so dass für jedes Kind X_i eine Färbung $c_i \in \text{Extcol}(X_i)$ existiert, wobei c und c_i für alle Knoten in $X \cap X_i$ übereinstimmen. Aus den Eigenschaften einer Baumzerlegung folgt, dass die Teilbäume T_1, \dots, T_m paarweise nur die Knoten teilen können, die auch in X enthalten sind. Deshalb können Färbungen $c_i \in \text{Extcol}(X_i)$, die jeweils mit c auf $X \cap X_i$ übereinstimmen, zusammen mit c zu einer Färbung zusammengefügt werden.

Der Algorithmus muss also für jede mögliche Kombination einer Färbung $c \in \text{Col}(X)$ und einer Färbung $c_i \in \text{Extcol}(X_i)$ überprüfen, ob sie für alle Knoten in $X \cap X_i$ übereinstimmen. Offensichtlich gilt $|\text{Extcol}(X_i)| \leq |\text{Col}(X_i)| \leq 3^{k+1}$ für jede Bag X_i . Somit gibt es $(3^{k+1})^2 = 3^{2(k+1)}$ mögliche Kombinationen von Färbungen. Die Überprüfung, ob c und c_i auf allen Knoten in $X \cap X_i$ übereinstimmen, benötigt höchstens Zeit $\mathcal{O}(k)$. Somit ist die Laufzeit des Algorithmus pro Kante der Baumzerlegung in $\mathcal{O}(3^{2(k+1)} \cdot k) = \mathcal{O}(3^{2k} \cdot k)$.

Mit einer etwas geschickteren Methode lässt sich diese Schranke sogar noch verbessern. Wir berechnen zuerst für jede Färbung $c_i \in \text{Extcol}(X_i)$ die Werte auf $X \cap X_i$, was in Zeit $\mathcal{O}(3^k \cdot k)$ machbar ist. Dann sortieren wir die Liste dieser Werte, was in Zeit $\mathcal{O}(3^k \cdot k) = \mathcal{O}(3^k \cdot k)$ machbar ist (etwa mit Radixsort). Dasselbe können wir mit den Färbungen $c \in \text{Col}(X)$ machen, ebenfalls in Zeit $\mathcal{O}(3^k \cdot k)$. Anschliessend können diese Listen parallel durchlaufen werden, um zu überprüfen, ob die Elemente in der Liste von $\text{Col}(X)$ auch in der Liste von $\text{Extcol}(X_i)$ erscheinen. Auch das ist in Zeit $\mathcal{O}(3^k \cdot k)$ möglich.

Diese Überprüfung wird auf jeder Kante der Baumzerlegung genau einmal ausgeführt (nämlich für den jeweiligen Elternknoten), insgesamt ist die Laufzeit des Algorithmus also in $\mathcal{O}(|E(T)| \cdot 3^k \cdot k)$. Wir gehen davon aus, dass die betrachtete Baumzerlegung einfach ist, also gilt gemäss Vorlesung, dass $|E(T)| = |V(T)| - 1 \leq |V(G)| - 1$. Die Gesamtlaufzeit des Algorithmus ist somit in $\mathcal{O}(|V(G)| \cdot 3^k \cdot k)$.