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 6

Zürich, 4. April 2023

Lösung zu Aufgabe 7

(a) Sei $C_{\max}$ eine Clique maximaler Grösse von G, $I_{\max}$ ein Independent-Set maximaler Grösse und sei (C, I) eine Split-Partition von G. Es kann höchstens ein Knoten von $C_{\max}$ in I sein. Umgekehrt kann C dann neben mindestens $|C_{\max}| - 1$ Knoten von $C_{\max}$ höchstens einen zusätzlichen Knoten enthalten, ansonsten wäre die Maximalität von $C_{\max}$ verletzt. Gleichzeitig kann höchstens ein Knoten von $I_{\max}$ in C sein und I enthält neben mindestens $|I_{\max}| - 1$ Knoten von $I_{\max}$ höchstens einen zusätzlichen Knoten. Der Graph G enthält also neben C' und I' höchstens zwei Knoten. Es gibt $|C_{\max}| + 1$ Möglichkeiten für die Wahl von $C \cap C_{\max}$ und $|I_{\max}| + 1$ Möglichkeiten für $C \cap I_{\max}$, sowie für jede dieser Aufteilungen höchstens zwei Möglichkeiten, die verbliebenen Knoten von G auf C und I aufzuteilen. Es gibt also höchstens $2 \cdot (|C_{\max}| + 1) \cdot (|I_{\max}| + 1) \in \mathcal{O}(|V|^2)$ Möglichkeiten, C zu bilden.
Wir halten ausserdem fest, dass für einen gegebenen Split-Graphen G in polynomieller Zeit alle Split-Partitionen erzeugt werden können: Für eine inklusionsmaximale Clique $C_{\max}$ müssen bis auf möglicherweise eine Ausnahme alle Knoten in C sein. Ausserdem können nur Knoten $x \in C_{\max}$ in I sein, für die $N[x] \subseteq C_{\max}$ gilt; Nachbarn von x könnten nicht in I sein, aber auch nicht in C ohne die Inklusionsmaximalität von $C_{\max}$ zu verletzen. Falls es keine solche Knoten gibt, muss $(C_{\max}, V \backslash C_{\max})$ also die einzige Split-Partition sein. Falls es mehrere solcher Knoten gibt, kann höchstens einer von ihnen in I sein und es können keine weiteren Knoten von G in C sein, wir überprüfen also höchstens $|C_{\max}| + 1$ mögliche Partitionen. Falls es genau einen solchen Knoten $x \in C_{\max}$ gibt, überprüfen wir, ob $(C_{\max}, V \backslash C_{\max})$ bereits eine Split-Partition ist. Anschliessend fügen wir x zu I hinzu, erweitern $C_{\max}$ zu einer inklusionsmaximalen Clique auf dem Graphen $G \backslash I$ und wiederholen den Prozess höchstens n mal.

(b) Wir betrachten eine Split-Partition (C_1, I_1) von $H[V \backslash T]$. Es kann höchstens ein Knoten v_c aus C_1 in I_X und höchstens ein Knoten v_i aus I_1 in C_X liegen. Wir werden alle dieser $\mathcal{O}(n^2)$ Möglichkeiten durchgehen. Wenn $C_0 \cup \{v_i\}$ keine Clique oder $I_0 \cup \{v_c\}$ kein Independent-Set sind, ignorieren wir diese Möglichkeit. Sei nun $\hat{I} = I_0 \cup I_1 \cup \{v_c\}$ und $\hat{C} = C_0 \cup C_1 \cup \{v_i\}$. Das sind gute Kandidaten für C_X und I_X. Dazu müssen wir aber sicher alle Knoten in $\hat{C}$, die nicht mit einem Knoten in $C_0 \cup \{v_i\}$ verbunden sind, entfernen, ebenso alle Knoten in $\hat{I}$, die mit einem Knoten in $C_1 \cup \{v_c\}$ verbunden sind. Bezeichnen wir mit X die Menge dieser Knoten. Wir finden nun entweder ein X mit $X \cap T = \emptyset$ und $|X| \leq r$ oder stellen fest, dass es keines gibt. Wenn wir aber ein solches X finden, ist es genau die Menge, die wir suchen. Die Partition $(\hat{C} \backslash X, \hat{I} \backslash X)$ ist eine Split-Partition. So ist $\hat{C} \backslash X$ sicher eine Clique: $C_0 \cup \{v_i\}$ ist eine Clique, C_1 ist eine Clique und jede Kante von C_1 nach $C_0 \cup \{v_i\}$ ist auch vorhanden, sonst hätten wir den Knoten in C_1 mit X entfernt. Aus demselben Grund ist auch $\hat{I} \backslash X$ ein Independent Set. Für jedes v_i und v_c ist diese Analyse in $\mathcal{O}(n)$ durchführbar, der ganze Prozess also in $\mathcal{O}(n^3)$.

(c) Stellen wir uns vor, wir hätten bereits eine solche Lösung S'. Dann betrachten wir die Teilmenge $Y = S' \cap S$. Dann muss der Graph $G[S \backslash Y]$ ein Split-Graph sein: Die Knoten in $S \backslash Y$ sind in $G[V \backslash S']$ enthalten, lassen sich also in eine Clique und ein Independent Set aufteilen. Wir wissen, dass die Knoten aus Y Teil unserer Lösung sein müssen. Betrachten wir nun den Graphen $H = G[V \backslash Y]$. Wenn wir aus H die Knoten in $S' \backslash Y$ auch noch entfernen, erhalten wir einen Split-Graph. Gleichzeitig bildet $S \backslash Y$ bereits eine Lösung für diesen Graphen. Schliesslich sind $S' \backslash Y$ und $S \backslash Y$ disjunkt. Wir können also mit $H = G[V \backslash Y]$ und $r = k - |Y|$ die Methode aus Teilaufgabe (c) anwenden. Im schlimmsten Fall müssen wir $\binom{k+1}{0} + \binom{k+1}{1} + \cdots + \binom{k+1}{k} \in \mathcal{O}(2^k)$ Teilmengen Y ausprobieren. Die entsprechende Split-Partition (C_0, I_0) probieren wir wieder aus, dafür gibt es nach Teilaufgabe (a) höchstens $\mathcal{O}(k^2)$ Möglichkeiten. Für jede dieser Instanzen benötigen wir nach Teilaufgabe (b) polynomiell viel Zeit.

(d) Sei $V = \{v_1, \ldots, v_n\}$. Dann berechnen wir nach und nach eine Lösung für die Instanzen $G[\{v_1, \ldots, v_i\}]$. Für $G_1 = G[\{v_1\}]$ ist offensichtlich $S_1 = \emptyset$ eine zulässige Lösung. Wenn wir eine Lösung S_i^* mit $|S_i^*| \leq k$ für den Graphen G_i berechnet haben, bildet offensichtlich $S_{i+1} = S_i^* \cup \{v_{i+1}\}$ eine Lösung für den Graphen G_{i+1} mit $|S_{i+1}| \leq k + 1$. Falls $|S_{i+1}| \leq k$, setzen wir $S_{i+1}^* = S_{i+1}$. Ansonsten berechnen wir mit Teilaufgabe (c) eine Lösung S_{i+1}^* für G_{i+1} mit $|S_{i+1}^*| \leq k$ oder erkennen, dass es keine solche Lösung für G_{i+1} und damit auch keine für G gibt. Die Lösung für die Instanz (G, k) bildet schliesslich S_n^*. Damit erhalten wir einen Algorithmus mit Laufzeit in $\mathcal{O}(2^k \cdot k^2 \cdot n^4)$.