Approximations- und Online-Algorithmen

Dr. Hans-Joachim Böckenhauer
Dr. Dennis Komm

courses.ite.inf.ethz.ch/approx_online_alg_22

Lösungsvorschläge – Blatt 2

Zürich, 10. März 2022

Lösung zu Aufgabe 2

(a) Man kann die Eingabeinstanzen folgendermassen umformen. Sei $G = (V, E)$ eine Eingabe für MinVC. Wir setzen $X = E$ und $\mathcal{F} = \{F_x \mid x \in V\}$, wobei $F_x = \{e \in E \mid x \in e\}$ für alle $x \in V$. Sei auf der anderen Seite $\mathcal{S} \subseteq \mathcal{F}$ eine Lösung für MinSC. Dann ist die entsprechende Lösung für MinVC gegeben durch die Menge $\{x \in V \mid F_x \in \mathcal{S}\}$.

Der Greedy-Algorithmus für die neue Instanz entspricht einem Greedy-Algorithmus für MinVC, bei dem in jedem Schritt der Knoten dem Vertex-Cover hinzugefügt wird, der am meisten noch nicht abgedeckte Kanten überdeckt.

Wendet man den Greedy-Algorithmus auf die neue Instanz an, hängt die Approximationsgüte von der maximalen Grösse der Mengen F_x ab, also von $\max_{x \in V}\{|F_x|\}$. Die Mächtigkeit der Menge F_x entspricht dem Grad des Knotens x in G. Der Greedy-Algorithmus ist damit ein $Har(\delta_{\max})$-Approximations-Algorithmus für MinVC, wobei $\delta_{\max}$ der maximale Knotengrad von G ist. Es gilt $Har(m) < 2$ für $m \leq 3$. Die gesuchten Spezialfälle sind demnach alle Graphen mit einem maximalen Knotengrad von 3.

(b) Der folgende Algorithmus löst TreeMinVC in Polynomzeit.

Eingabe: $G = (V, E)$, G ist ein Baum.

1. $S := \emptyset$, $V' := V$ und $E' := E$.
2. **while** $E' \neq \emptyset$
   Bestimme alle isolierten Kanten und füge jeweils einen ihrer inzidenten Knoten zu S hinzu, lösche diese Knoten aus V' und die Kanten aus E'. Bestimme alle Blattknoten (Knoten vom Grad 1) aus V' und füge die Nachbarknoten dieser Blattknoten der Menge S hinzu. Lösche alle überdeckten Kanten aus E' und die nun isolierten Knoten aus V'.

Ausgabe: Die Knotenmenge S.

Der Algorithmus terminiert auf jeden Fall, da in jedem Schritt mindestens eine Kante aus E' gelöscht wird. Damit wird die **while**-Schleife höchstens $|E|$-mal durchgeführt, die Laufzeit des Algorithmus ist sogar linear in der Eingabelänge, da jede Kante in der Eingabe enthalten sein muss. Die Ausgabe ist auch immer eine zulässige Lösung, da nur überdeckte Kanten aus E' entfernt werden. Nun kann man sich überlegen, dass die Lösung des Algorithmus optimal ist. Es ist klar, dass jede Kante zu einem Blattknoten entweder vom Blattknoten oder von dessen Nachbarn überdeckt werden kann. Da ein Blattknoten nur eine Kante abdeckt, ist es in jedem Fall sinnvoller, die Nachbarn von Blattknoten einem minimalen Vertex-Cover hinzuzufügen. Damit folgt, dass TreeMinVC polynomiell optimal lösbar ist.

Lösung zu Aufgabe 3

(a) Wir geben einen Greedy-Algorithmus zum Lösen von MaxSAT an.

Eingabe: CNF-Formel $\Phi = C_1 \wedge \cdots \wedge C_m$ über $X = \{x_1, \ldots, x_n\}$.

1. $X' := X$, $\Phi' := \Phi$.
2. **while** $\Phi' \neq \emptyset$
   Suche die Variable $x_i \in X'$, so dass eine Variablenbelegung $\alpha(x_i)$ möglichst viele Klauseln erfüllt. Füge x_i mit dieser Belegung zu α hinzu, lösche x_i aus X' und lösche alle Klauseln aus Φ', deren Wahrheitswert durch die Belegung von x_i festgelegt ist.
3. Erweitere die Variablenbelegung α für alle Variablen, die noch nicht belegt sind, um eine Belegung mit 0.

Ausgabe: Die Belegung α.

Die **while**-Schleife wird maximal $|\Phi|$-mal durchlaufen und in jedem Schleifendurchlauf sind nur polynomiell viele Schritte durchzuführen. Damit terminiert der Algorithmus immer in Polynomzeit. Die Lösung ist immer eine Variablenbelegung. Die Variablen, die in Schritt 2 nicht belegt werden, können beliebig belegt werden, in unserem Fall alle mit 0.

(b) Für MaxE2SAT gibt es für jede Klausel zwei Chancen, diese Klausel zu erfüllen. Wird eine Klausel C_i mit den Literalen l_1 und l_2 nicht erfüllt, ist keines der beiden Literale erfüllt. Wird die Variablenbelegung für l_1 so gewählt, dass C_i dadurch noch nicht erfüllt ist, wird mindestens eine andere Klausel erfüllt. Da die bereits erfüllten Klauseln bei der Greedy-Auswahl der Belegung der übrigen Variablen nicht berücksichtigt werden, gilt dasselbe für die Variablenbelegung für l_2. Also werden für jede nicht erfüllte Klausel mindestens 2 andere Klauseln erfüllt. Somit ergibt sich ein Verhältnis von 2 erfüllten Klauseln auf 3 Klauseln in der Formel. Somit erreicht der Greedy-Algorithmus eine Approximationsgüte von 3/2.