BuK Abgabe 11 | Gruppe 17

Malte Meng (354529) , Charel Ernster (318949), Sebastian Witt (354738)

January 19, 2017

1 Aufgabe 11.1

Sei eine Instanz von 3-SAT gegeben, also eine aussagenlogische Formel ϕ in 3-KNF, mit der Frage, ob es eine erfüllende Belegung α für ϕ existiert.
Nun wollen wir daraus eine Instanz von NOT-ALL-EQUAL-SAT erzeugen. Dazu transformieren wir ϕ zu einer neuen aussagenlogische Formel ψ in 3-KNF, indem wir jede Klausel in ϕ in zwei neue Klauseln umwandeln:

$$(x_i \vee x_j \vee x_k) \rightarrow (x_i \vee x_j \vee y_{i,j}) \wedge (\neg y_{i,j} \vee x_k \vee x_l)$$

wobei wir später noch über die Rolle von x_l sprechen wollen. Diese Transformation ist in polynomieller Zeit gemessen an der Eingabelänge möglich, da lediglich pro Klausel in einem einfachen Verfahren eine zweite Klausel erstellt wird, welche eine neue Variable $y_{i,j}$ und die wiederbenutzte Variable x_l einführt.
Es bleibt zu zeigen, dass eine solche NOT-ALL-EQUAL-SAT-Instanz genau dann eine Ja-Instanz ist, wenn bereits die Ausgangsinstanz von 3-SAT eine Ja-Instanz ist. Dazu nehmen wir an, dass die 3-SAT-Instanz ϕ eine Ja-Instanz ist und so eine erfüllende Belegung α besitzt. Dann erhalten wir eine erfüllende Belegung β für ψ, welche die Zusatzbedingung von NOT-ALL-EQUAL-SAT erfüllt, indem wir alle übernommenen Variablen x_i einfach $\beta(x_i) = \alpha(x_i)$ setzen. Zudem setzen wir $\beta(x_l) = 0$. Entsprechend erhalten wir also für unsere Klauseln mehrere Möglichkeiten:

- $\alpha(x_i) = \alpha(x_j) = 1$ gilt. Dann müssen wir $\beta(y_{i,j}) = 0$ setzen und entsprechend ist die zweite erzeugte Klausel ebenfalls erfüllt. Da $\beta(x_l) = 0$ ist, sind die Bedingungen von NOT-ALL-EQUAL-SAT für die Klauseln erfüllt. β erfüllt die Klauseln und in jeder Klausel gibt es ein falsches Literal.
- $\alpha(x_i) \neq \alpha(x_j)$ gilt. Dann ist die erste erfüllt und wir können $\beta(y_{i,j})$ nach Belieben setzen. Wir setzen nun $\beta(y_{i,j}) = \alpha(x_k)$ und somit ist bereits $(\neg y_{i,j} \vee x_k)$ stets erfüllt und die NOT-ALL-EQUAL-SAT-Zusatzbedingung ebenfalls.

- $\alpha(x_i) = \alpha(x_j) = 0$ gilt. Dann müssen wir $\beta(y_{i,j}) = 1$ setzen, um die erste Klausel zu erfüllen. In der zweiten Klausel muss aber $\alpha(x_k) = 1$ bereits gelten, da α ein Modell für ϕ ist. Mit $\beta(\neg y_{i,j}) = 0$ folgt also sofort, dass auch die zweite Klausel erfüllt ist und der Zusatzbedingung genügt.

Entsprechend ist β ein Modell für ψ, welches zudem die Nebenbedingung von NOT-ALL-EQUAL-SAT erfüllt und somit ist ψ eine Ja-Instanz von NOT-ALL-EQUAL-SAT.

Andersherum: Sei ψ eine Ja-Instanz von NOT-ALL-EQUAL-SAT erfüllt, existiert also ein Modell β für ψ gegeben, welches die Nebenbedingung von NOT-ALL-EQUAL-SAT erfüllt ist. Nun erhalten wir zwei Fälle:

- $\beta(x_l) = 0$ gilt. Dann muss für mindestens ein Literal x_i aus ϕ aus jedem Klauselpaar, welches aus einer Klausel von ϕ konstruiert wurde, $\beta(x_i) = 1$ gelten. Entsprechend ist jede Klausel von ϕ erfüllt durch die Belegung $\beta|_{x_i \in \phi}$ und somit ist ϕ eine Ja-Instanz von 3-SAT.
- $\beta(x_l) = 1$ gilt. Dann können wir die Belegung einfach invertieren, da die Invertierung einer erfüllenden NOT-ALL-EQUAL-SAT-Belegung wiederum eine erfüllende NOT-ALL-EQUAL-SAT-Belegung ist (Es gab stets mindestens ein unerfülltes Literal und ein erfülltes. Invertieren wir die Belegung, so ist das vorher unerfüllte Literal nun erfüllt und das vorher Erfüllte nun unerfüllt) und wir erhalten für die invertierte Belegung $\bar{\beta}$ den ersten Fall.

Entsprechend muss, falls die erzeugte Formel ψ eine Ja-Instanz von NOT-ALL-EQUAL-SAT ist, bereits die Ausgangsinstanz von 3-SAT eine Ja-Instanz gewesen sei.

Insgesamt gilt also 3-SAT$\leq_p$ NOT-ALL-EQUAL-SAT.

2 Aufgabe 11.2

2-Approximation für VertexCover:
Für Graph G = (V,E).

1. Wähle einen Knoten v_{max} mit den meisten inzidenten Kanten E_{sub}.
2. Wiederhole dies für den Teilgraphen G_{sub} = (V/v_{max},E/E_{sub}) bis $|E| = 0$

Die gewählten Knoten entsprechen einem VertexCover mit $|C| = 2 * min$VertexCover.
Dieser Algorithmus ist offensichtlich, da die Iteration nach linearer Zeit terminiert. Abhängig von der Kantenmenge. **Korrektheit:**
Es wird offensichtlich ein VertexCover berechnet, da nach der letzten Iteration der Teilgraph keine Kanten mehr enthält. Für die Entscheidung ob ein Knoten im VertexCover ist, ist nur die direkte Nachbarschaft wichtig.

Denn bei Verbindungen über zwei Kanten braucht es mindestens den ersten Knoten und den indirekt verbundenen oder den zweiten in der Mitte. Wenn nun zunächst die Knoten mit den meisten Kanten gewählt werden kommt es zu zwei Möglichkeiten:

1. es gibt den Knoten mit den meisten Kanten. Durch "Falschauswahl" kann maximal eine Kante in der Nachbarschaft gewählt werden. Da aber der Grad sowieso mindestens größer, als der der Nachbarn ist, ist der Knoten minimal.

2. es gibt mehrere Knoten mit maximalem grad. Durch "Falschauswahl" kann eine Kante in der Nachbarschaft ungünstig gewählt werden, sodass ein Knoten mehr als die optimale Lösung gewählt wird.

Es kommt also maximal pro "Falschauswahl" zu doppelter Knotenmenge. Wird bei jedem Knoten ungünstig ausgewählt, ist die Lösung im worst-case doppelt so groß wie die Minimale. Es ist also eine gültige 2-Aproximation von Vertex-Cover.