automatisierte Laufzeitanalyse

# Algorithmen und Programmierung 3, WS 2019/2020 — 12. Übungsblatt

Abgabe bis Freitag, 24. Januar 2020, 12:00 Uhr, in die Fächer der Tutoren

77. Polynomielle Reduktion, kürzester Weg mit negativen Kantengewichten, 6 Punkte

Das Problem $A$ sei das Problem, in einem gerichteten Graphen mit *beliebigen* ganzzahligen Kantengewichten einen kürzesten Weg von $s$ nach $t$ zu finden, der keinen Knoten mehrfach besucht (einen einfachen Weg). Das Problem $B$ sei das Problem, zu überprüfen, ob ein ungerichteter Graph einen *Hamilton-Kreis* enthält, das heißt, einen Kreis, der jeden Knoten *genau einmal* durchläuft.

Zeigen Sie: $B <_P A$.

78. Konjunktive Normalform, 0 Punkte

Schreiben Sie Boolesche Formeln in konjunktiver Normalform für folgende Aussagen. Versuchen Sie, mit möglichst wenigen Klauseln auszukommen.

(a) Höchstens eine der Variablen $x_1, \ldots, x_5$ ist wahr.

(b) Mindestens zwei der Variablen $x_1, \ldots, x_5$ sind wahr.

(c) Keine zwei benachbarten Variablen $x_i$ und $x_{i+1}$ von den Variablen $x_1, \ldots, x_5$ sind gleichzeitig falsch.

(d) Die durch die Bits $(x_4 \ldots x_0)$ gegebene Binärzahl ist kleiner als elf.

79. Erfüllbarkeit (SAT), Programmieraufgabe, 8 Punkte

Eine Boolesche Formel $\phi(x_1, ..., x_n)$ in konjunktiver Normalform ist genau dann erfüllbar, wenn $\phi(1, x_2, ..., x_n)$ oder $\phi(0, x_2, ..., x_n)$ erfüllbar ist. Dabei ensteht die Formel $\phi(1, x_2, ..., x_n)$ aus $\phi(x_1, ..., x_n)$, indem jedes Vorkommen von $x_1$ durch 1 ersetzt wird. Das Ergebnis kann vereinfacht werden: Jede Klausel, die $x_1$ enthält, kann weggelassen werden. Steht das Literal $\bar{x}_1$ in einer Klausel, kann man es streichen. Für die Formel

$$\phi(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3) \wedge \bar{x}_3$$

vereinfacht sich zum Beispiel $\phi(1, x_2, x_3)$ zu $(\bar{x}_2 \vee x_3) \wedge \bar{x}_3$.

Analoges gilt für $\phi(0, x_2, ..., x_n)$. Eine konjunktive Normalform, die keine Klauseln mehr enthält („leere Konjunktion“) ist erfüllbar; eine konjunktiver Normalform, die eine leere Klausel enthält („leere Disjunktion“), ist nicht erfüllbar.

Implementieren Sie einen entprechenden rekursiven Entscheidungsalgorithmus für SAT in Python oder Java, der auch gegebenfalls eine erfüllende Belegung ausgibt. Zählen Sie in geeigneter Form die Schritte des Programmes, zum Beispiel die Anzahl der rekursiven Aufrufe. Die Eingabeformel in konjunktiver Normalform soll wie im folgenden Beispiel codiert sein: $(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_3) \wedge \bar{x}_2$ als `1 2 -3, -1 3, -2.`

Testen Sie Ihr Programm zum Beispiel mit der Konjunktion aus Aufgabe 78b und 78d.

80. NP, 6 Punkte. Zeigen Sie, dass die folgenden Probleme in NP sind, und geben Sie jeweils das Zertifikatskriterium an.

(a) *Eingabe:* Eine natürliche Zahl $k$ in Binärdarstellung
*Frage:* Ist $k$ zusammengesetzt? (das heißt, keine Primzahl)
*Anmerkung:* Die Multiplikation oder Division zweier $n$-stelliger Zahlen nach der Schulmethode erfordert $O(n^2)$ Zeit.

(b) *Eingabe:* Eine Menge von $n$ Paketen mit Gewichten $w_1, \ldots, w_n \in \mathbb{N}$ und zwei Zahlen $k, v \in \mathbb{N}$.
*Frage:* Lassen Sich die Pakete mit $k$ Autos transportieren, wenn jedes Auto höchstens Pakete mit Gesamtgewicht $v$ laden kann?

77. Polynomielle Reduktion, kürzester Weg mit negativen Kantengewichten, 6 Punkte

Das Problem $A$ sei das Problem, in einem gerichteten Graphen mit *beliebigen* ganzzahligen Kantengewichten einen kürzesten Weg von $s$ nach $t$ zu finden, der keinen Knoten mehrfach besucht (einen einfachen Weg). Das Problem $B$ sei das Problem, zu überprüfen, ob ein ungerichteter Graph einen *Hamilton-Kreis* enthält, das heißt, einen Kreis, der jeden Knoten *genau einmal* durchläuft.

Zeigen Sie: $B <_P A$. 3-Fkt Problem entsteht Hamilton-Kreis auf Rundreise

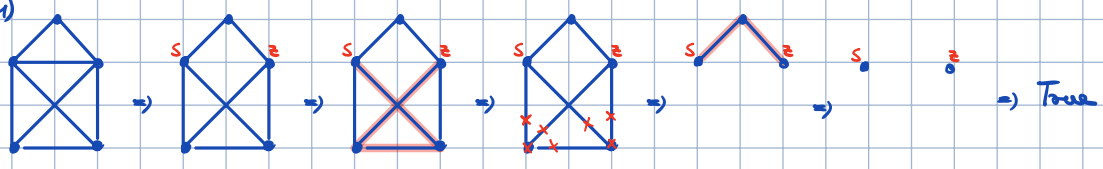
n Der Graph ist ungerichtet
1 Wähle einen beliebigen Startpunkt.
1 Wähle das Ziel abhängig vom Startpunkt mit einer Kante Entfernung von eins. Lösche diese Kante
n Lege das Kantengewicht für alle Kanten auf -1

$n^{c+1}$ While ( Findet Kürzesten Weg == True und Knoten $\neq 2$ ):
- $n^c$ Finde Kürzesten Weg und lösche alle Knoten dieses Graphen, außer vom Start und Zielpunkt.
- n lösche alle Kanten der gelöschten Knoten

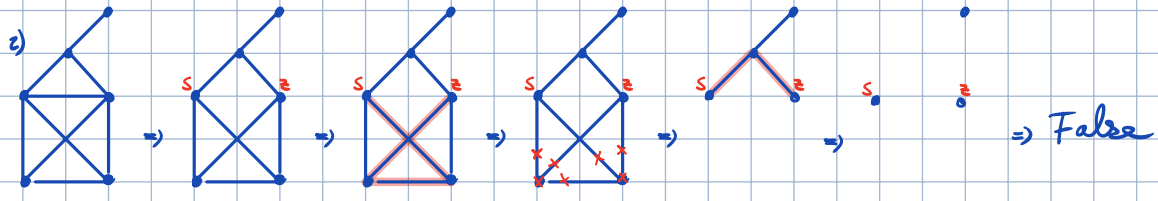
1 noch Knoten == 2

$\Rightarrow n^{c+1}$ liegt in NP

Bsp 1)




2)



80. NP, 6 Punkte. Zeigen Sie, dass die folgenden Probleme in NP sind, und geben Sie jeweils das Zertifikatskriterium an.

NP = überprüfbar in $(n^c)$

(a) *Eingabe:* Eine natürliche Zahl $k$ in Binärdarstellung
*Frage:* Ist $k$ zusammengesetzt? (das heißt, keine Primzahl)
*Anmerkung:* Die Multiplikation oder Division zweier $n$-stelliger Zahlen nach der Schulmethode erfordert $O(n^2)$ Zeit.

(b) *Eingabe:* Eine Menge von $n$ Paketen mit Gewichten $w_1, \ldots, w_n \in \mathbb{N}$ und zwei Zahlen $k, v \in \mathbb{N}$.
*Frage:* Lassen Sich die Pakete mit $k$ Autos transportieren, wenn jedes Auto höchstens Pakete mit Gesamtgewicht $v$ laden kann?

a)

```
n/2   for i in range(2, ⌈k/2⌉):
2n²     if( k/i == ⌈k/i⌉ ): return True
        :
1     return False
```

$\Rightarrow \frac{n}{2} \cdot 2n^2$, $n^3$ liegt in NP

b) n Pakete mit $w_1, \ldots w_n$ Gewichten
k Autos mit v Platz

```
1   platz = v
n   for paket in (w1, w2, ..., wn):
1     if (k == 0): return False

n     if((platz - paket) > 0): platz -= paket

1     else: k -= 1, platz = v
```

$\Rightarrow n^2$, n liegt in NP

# 79. Erfüllbarkeit (SAT), Programmieraufgabe, 8 Punkte

```
thob97@andorra:~/alp3/u12$ python3 u12.py
Eingabe: [[1, 2, 3], [1, -2], [-1, -2, 3], [-3]] Ergebnis: [1, -2, -3] Rekursionen: 13
Eingabe: [[1, 2, -3, -4, -5], [-5], [-4], [-3, -4]] Ergebnis: [-4, -5] Rekursionen: 48
Eingabe: [[-1], [1]] Ergebnis: None Rekursionen: 2
thob97@andorra:~/alp3/u12$
```

mehr in code