Theoretische Informatik

FSS 2020 - Tutorium #3

Alexander Moch

12. Mai 2020

Aufgabe 3.1

Aufgabe 3.1

Zeige, dass $\leq_p$ eine reflexive und transitive Relation auf $\mathcal{NP}$ ist.

Hinweis: O.B.d.A. kann hier angenommen werden, dass die Polynome, welche die Worst-Case-Rechenzeiten der entsprechenden Transformationen angeben, monoton wachsend seien.

Aufgabe 3.1

Um **Transitivität** zu beweisen, muss gezeigt werden, dass

$$((L_1 \leq_p L_2) \wedge (L_2 \leq_p L_3)) \Rightarrow L_1 \leq_p L_3.$$

Aufgabe 3.1

Nach Voraussetzung gibt es Funktionen

$$f_1 : \Sigma_1^* \to \Sigma_2^* \text{ und}$$
$$f_2 : \Sigma_2^* \to \Sigma_3^*,$$

die in polynomieller Worst-Case-Zeit p_1 bzw. p_2 berechnet werden können und für die gilt:

$$\forall x \in \Sigma_1^* : x \in L_1 \Leftrightarrow f_1(x) \in L_2 \text{ und}$$
$$\forall y \in \Sigma_2^* : y \in L_2 \Leftrightarrow f_2(y) \in L_3.$$

Aufgabe 3.1

- O.B.d.A. sei p_2 monoton wachsend.

Aufgabe 3.1

- O.B.d.A. sei p_2 monoton wachsend.
- Wir definieren $f_3 : \Sigma_1^* \to \Sigma_3^*$ als $f_3 := f_2 \circ f_1$.

Aufgabe 3.1

- O.B.d.A. sei p_2 monoton wachsend.
- Wir definieren $f_3 : \Sigma_1^* \to \Sigma_3^*$ als $f_3 := f_2 \circ f_1$.
- Dann ist, da $|f_1(x)| \leq p_1(|x|)$, f_3 in polynomieller Worst-Case-Zeit $p_1 + p_2 \circ p_1$ berechenbar und es gilt:

$$\forall x \in \Sigma_1^* : x \in L_1 \Leftrightarrow f_1(x) \in L_2 \Leftrightarrow f_2 \circ f_1(x) \in L_3 \Leftrightarrow f_3(x) \in L_3.$$

Aufgabe 3.1

Um **Reflexivität** zu beweisen, wählen wir

$$f := id.$$

Aufgabe 3.2

Aufgabenstellung + Lösung

Man zeige, dass $\mathcal{NP}$ abgeschlossen gegenüber $\leq_p$ ist (d.h. aus $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$ folgt $L_1 \in \mathcal{NP}$).

Lösung

Wir schreiben die Voraussetzung ausführlich:

[1] p_M ist ein Polynom; n ist die Länge der Eingabe für M bzw. f.

Lösung

Wir schreiben die Voraussetzung ausführlich:

- Es gibt einen Polynomialzeitalgorithmus M, der in Worst-Case-Zeit $p_M(n)$[1] eine Funktion $f : \Sigma_1^* \to \Sigma_2^*$ berechnet, welche $w \in L_1 \Leftrightarrow f(w) \in L_2$ erfüllt.

[1] p_M ist ein Polynom; n ist die Länge der Eingabe für M bzw. f.

Lösung

Wir schreiben die Voraussetzung ausführlich:

- Es gibt einen Polynomialzeitalgorithmus M, der in Worst-Case-Zeit $p_M(n)$[1] eine Funktion $f : \Sigma_1^* \to \Sigma_2^*$ berechnet, welche $w \in L_1 \Leftrightarrow f(w) \in L_2$ erfüllt.
- Außerdem gibt es einen PRV-Algorithmus N_2, der für $z \in \Sigma_2^*$ in polynomieller Worst-Case-Zeit $p_2(|z|)$ entscheidet, ob $z \in L_2$ ist.

[1] p_M ist ein Polynom; n ist die Länge der Eingabe für M bzw. f.

Lösung - Fortsetzung

Wir beschreiben nun einen PRV-Algorithmus N_1 für L_1.

Lösung - Fortsetzung

Wir beschreiben nun einen PRV-Algorithmus N_1 für L_1.

- Für die Eingabe w wird zunächst ein Unterprogramm verwendet, das M in Polynomialzeit auf w ausführt und $f(w)$ berechnet.

Lösung - Fortsetzung

Wir beschreiben nun einen PRV-Algorithmus N_1 für L_1.

- Für die Eingabe w wird zunächst ein Unterprogramm verwendet, das M in Polynomialzeit auf w ausführt und $f(w)$ berechnet.
- Dann wird in ein zweites Unterprogramm gewechselt, das N_2 auf $f(w)$ ausführt und entscheidet, ob $f(w) \in L_2$.

Lösung - Fortsetzung

Wir beschreiben nun einen PRV-Algorithmus N_1 für L_1.

- Für die Eingabe w wird zunächst ein Unterprogramm verwendet, das M in Polynomialzeit auf w ausführt und $f(w)$ berechnet.
- Dann wird in ein zweites Unterprogramm gewechselt, das N_2 auf $f(w)$ ausführt und entscheidet, ob $f(w) \in L_2$.
- Nach Voraussetzung ist damit entschieden, ob $w \in L_1$ ist.

Lösung - Fortsetzung

Wir beschreiben nun einen PRV-Algorithmus N_1 für L_1.

- Für die Eingabe w wird zunächst ein Unterprogramm verwendet, das M in Polynomialzeit auf w ausführt und $f(w)$ berechnet.
- Dann wird in ein zweites Unterprogramm gewechselt, das N_2 auf $f(w)$ ausführt und entscheidet, ob $f(w) \in L_2$.
- Nach Voraussetzung ist damit entschieden, ob $w \in L_1$ ist.

Da $p_M(|w|)$ polynomiell in $|w|$ beschränkt ist, gilt dies auch für $|f(w)|$. Daher ist die Worst-Case-Rechenzeit des zweiten verwendeten Unterprogramms ebenfalls polynomiell in $|w|$ beschränkt. Folglich entscheidet N_1 in polynomieller Zeit die Sprache L_1. □

Aufgabe 3.3

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Zeige, dass die folgenden Probleme (als Sprachen formuliert) polynomiell isomorph sind.

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Zeige, dass die folgenden Probleme (als Sprachen formuliert) polynomiell isomorph sind.

Die Instanzen bestehen aus einem (ungerichteten) Graphen $G = (V, E)$ und einer Zahl $K \in \{0, \ldots, |V|\}$.

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Zeige, dass die folgenden Probleme (als Sprachen formuliert) polynomiell isomorph sind.

Die Instanzen bestehen aus einem (ungerichteten) Graphen $G = (V, E)$ und einer Zahl $K \in \{0, \ldots, |V|\}$.

Independent Set : G enthält eine unabhängige Menge der Größe mindestens K.

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Zeige, dass die folgenden Probleme (als Sprachen formuliert) polynomiell isomorph sind.

Die Instanzen bestehen aus einem (ungerichteten) Graphen $G = (V, E)$ und einer Zahl $K \in \{0, \ldots, |V|\}$.

Independent Set : G enthält eine unabhängige Menge der Größe mindestens K.

Clique : G einhält eine Clique der größe mindestens K.

Aufgabenstellung

Wir nennen zwei Sprachen $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ polynomiell isomorph, wenn es eine bijektive Abbildung $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass f eine polynomielle Reduktion von L_1 nach L_2 und f^{-1} eine polynomielle Reduktion von L_2 nach L_1 ist.

Zeige, dass die folgenden Probleme (als Sprachen formuliert) polynomiell isomorph sind.

Die Instanzen bestehen aus einem (ungerichteten) Graphen $G = (V, E)$ und einer Zahl $K \in \{0, \ldots, |V|\}$.

Independent Set : G enthält eine unabhängige Menge der Größe mindestens K.

Clique : G einhält eine Clique der größe mindestens K.

Vertex Cover : G enthält eine Knotenüberdeckung der Größe höchstens K.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

II) W ist eine Clique im Komplementärgraphen G^c von G, wobei $G^c := (V, E^c)$, $E^c := \{\{u, v\} \subseteq V | u \neq v, \{u, v\} \notin E\}$.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

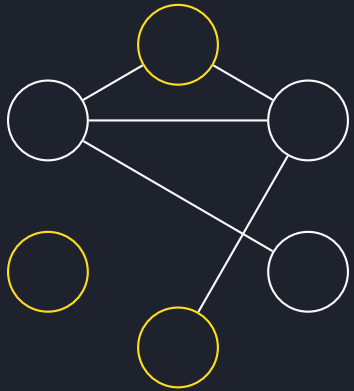
I) W ist eine unabhängige Menge in G.

II) W ist eine Clique im Komplementärgraphen G^c von G, wobei $G^c := (V, E^c)$, $E^c := \{\{u, v\} \subseteq V | u \neq v, \{u, v\} \notin E\}$.

III) $V \backslash W$ ist eine Knotenüberdeckung für G.

Beispiel für Äquivalenz von (I) Indendent Set und (II) Clique:

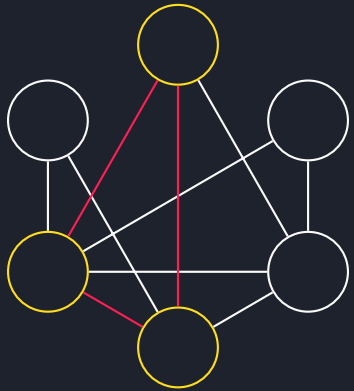
(I) Ein Independent Set[2] (gelbe Knoten) in *G*



[2] Das Independent Set ist hier nicht maximal.

Beispiel für Äquivalenz von (I) Independent Set und (II) Clique:

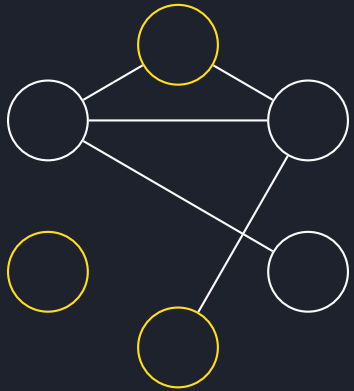
(II) Die entsprechende Clique[3] (gelbe Knoten) in G^c



[3] Die Clique ist hier nicht maximal.

Beispiel für Äquivalenz von (I) Independent Set und (III) Vertex Cover:

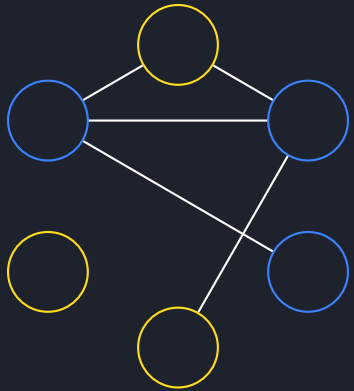
(I) Ein Independent Set[4] (gelbe Knoten) in *G*



[4]Das Independent Set ist hier nicht maximal.

Beispiel für Äquivalenz von (I) Independent Set und (III) Vertex Cover:

(III) Das entsprechende Vertex Cover[5] (blaue Knoten) in *G*



[5] Das Vertex Cover ist hier nicht minimal.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

II) W ist eine Clique im Komplementärgraphen G^c von G, wobei $G^c := (V, E^c)$, $E^c := \{\{u, v\} \subseteq V | u \neq v, \{u, v\} \notin E\}$.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

II) W ist eine Clique im Komplementärgraphen G^c von G, wobei $G^c := (V, E^c)$, $E^c := \{\{u, v\} \subseteq V | u \neq v, \{u, v\} \notin E\}$.

III) $V \backslash W$ ist eine Knotenüberdeckung für G.

Lösung

Man beobachtet leicht, dass die folgenden Aussagen äquivalent sind:

I) W ist eine unabhängige Menge in G.

II) W ist eine Clique im Komplementärgraphen G^c von G, wobei $G^c := (V, E^c)$, $E^c := \{\{u, v\} \subseteq V | u \neq v, \{u, v\} \notin E\}$.

III) $V \backslash W$ ist eine Knotenüberdeckung für G.

Nun sind die Transformationen klar:

- $f_{I \to II}$ (Independent Set auf Clique) muss (G, K) auf (G^c, K) abbilden.
- $f_{I \to III}$ (Independent Set auf Vertex Cover) muss (G, K) auf $(G, |V| - K)$ abbilden.
- Beide Abbildungen sind bijektiv und in polynomieller Zeit berechenbar.
- $f_{II \to III}$ (Clique auf Vertex Cover) ergibt sich durch Komposition:

$$f_{II \to III} := f_{I \to III} \circ f_{I \to II}^{-1}.$$

Aufgabe 3.4

Aufgabenstellung

In dieser Aufgabe sind die Eingaben DNF-Formeln (d.h. Disjunktionen $M_1 \vee \cdots \vee M_n$ von Konjunktionen $L_1 \wedge \cdots \wedge L_s$ von Literalen L_i). Für jedes der folgenden Probleme ist zu entscheiden, ob es in $\mathcal{P}$ liegt oder $\mathcal{NP}$-schwer ist (jeweils mit Begründung).

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

b) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Null macht?

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Dieses Problem ist in $\mathcal{P}$.

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Dieses Problem ist in $\mathcal{P}$. Ein Monom ist genau dann erfüllbar, wenn

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Dieses Problem ist in $\mathcal{P}$. Ein Monom ist genau dann erfüllbar, wenn es nicht eine Variable und ihre Negation enthält.

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Dieses Problem ist in $\mathcal{P}$. Ein Monom ist genau dann erfüllbar, wenn es nicht eine Variable und ihre Negation enthält. Ein Polynom (also eine DNF-Formel) ist genau dann erfüllbar, wenn

Lösung - a)

a) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Eins macht?

Lösung:

Dieses Problem ist in $\mathcal{P}$. Ein Monom ist genau dann erfüllbar, wenn es nicht eine Variable und ihre Negation enthält. Ein Polynom (also eine DNF-Formel) ist genau dann erfüllbar, wenn mindestens eines seiner Monome erfüllbar ist.

Beispiel für zwei Variablen x_1 und x_2:

(	x_1	$\wedge$	$\neg x_1$	)	$\vee$	(	x_1	$\wedge$	x_2	)	$\vee$	(	$\neg x_1$	$\wedge$	$\neg x_2$	)
(	1	$\wedge$	0	)	$\vee$	(	1	$\wedge$	1	)	$\vee$	(	0	$\wedge$	0	)

Lösung - b)

b) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Null macht?

Lösung:

Lösung - b)

b) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Null macht?

Lösung:

Dieses Problem ist $\mathcal{NP}$-vollständig.

Lösung - b)

b) Gibt es eine Belegung der Variablen, welche die DNF-Formel zu Null macht?

Lösung:

Dieses Problem ist $\mathcal{NP}$-vollständig. Jede Instanz von SAT kann durch Negation und Anwendung DeMorgan'scher Regeln in eine Instanz dieses Problems umgewandelt werden.

$$
\begin{aligned}
1 &= (\neg x_1 \vee x_1) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2) \\
\Leftrightarrow 0 &= \neg((\neg x_1 \vee x_1) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2)) \\
&= \neg(\neg x_1 \vee x_1) \vee \neg(\neg x_1 \vee \neg x_2) \vee \neg(x_1 \vee x_2) \\
&= (x_1 \wedge \neg x_1) \vee (x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2)
\end{aligned}
$$

Aufgabe 3.5

Aufgabenstellung

a) Finde eine polynomielle Reduktion von HC zu DHC.

b) Finde eine polynomielle Reduktion von DHC zu HC.

Lösung - a)

Um zu zeigen, dass $HC \leq_p DHC$, müssen wir lediglich jede ungerichtete Kante $\{u, v\}$ im HC Problem durch zwei gerichtete Kanten $(u, v), (v, u)$ im DHC ersetzen.

Lösung - b)

Wir zeigen $DHC \leq_p HC$.

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Ein gerichteter Hamiltonkreis im gegebenen Graphen lässt sich sofort in einen ungerichteten Hamiltonkreis im neuen Graphen übersetzen.

Lösung - b)

Wir zeigen $DHC \leq_p HC$.

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Wenn im neuen Graphen ein ungerichteter Hamiltonkreis gegeben ist, haben wir zwei mögliche ”Richtungen”, ihn zu durchlaufen.

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Wenn im neuen Graphen ein ungerichteter Hamiltonkreis gegeben ist, haben wir zwei mögliche ”Richtungen”, ihn zu durchlaufen.
- Wir legen eine Richtung fest, indem wir für eine beliebige Kante die Richtung wählen, die ihr im gerichteten Graphen entspricht.

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Wenn im neuen Graphen ein ungerichteter Hamiltonkreis gegeben ist, haben wir zwei mögliche ”Richtungen”, ihn zu durchlaufen.
- Wir legen eine Richtung fest, indem wir für eine beliebige Kante die Richtung wählen, die ihr im gerichteten Graphen entspricht.
- Wir erhalten dann direkt einen gerichteten Hamiltonkreis.

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Wenn im neuen Graphen ein ungerichteter Hamiltonkreis gegeben ist, haben wir zwei mögliche ”Richtungen”, ihn zu durchlaufen.
- Wir legen eine Richtung fest, indem wir für eine beliebige Kante die Richtung wählen, die ihr im gerichteten Graphen entspricht.
- Wir erhalten dann direkt einen gerichteten Hamiltonkreis.
- Würden wir nämlich in der obigen Abbildung den linkesten Knoten in der ungerichteten Komponente von links erreichen und nach links verlassen,

Lösung - b)

Wir zeigen $DHC \leq_p HC$. Dafür genügt folgende lokale Ersetzung:

- Wenn im neuen Graphen ein ungerichteter Hamiltonkreis gegeben ist, haben wir zwei mögliche ”Richtungen”, ihn zu durchlaufen.
- Wir legen eine Richtung fest, indem wir für eine beliebige Kante die Richtung wählen, die ihr im gerichteten Graphen entspricht.
- Wir erhalten dann direkt einen gerichteten Hamiltonkreis.
- Würden wir nämlich in der obigen Abbildung den linkesten Knoten in der ungerichteten Komponente von links erreichen und nach links verlassen, wäre der mittlere Knoten nicht mehr auf einem Hamiltonkreis passierbar. □


That's all Folks!