

11c

\mathcal{NP} -Vollständigkeit

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik und Theorembeweisen

Stand: 10. Mai 2024

Basiert auf Folien von PD Dr. David Sabel



Definition

Für eine Funktion $f : \mathbb{N} \rightarrow \mathbb{N}$ sei die Klasse $TIME(f(n))$ genau die Menge der Sprachen L , für die es eine stets anhaltende Mehrband-DTM M gibt mit $L(M) = L$ und $time_M(w) \leq f(|w|)$ für alle $w \in \Sigma^*$.

Definition

Die Klasse \mathcal{P} ist definiert als

$$\mathcal{P} := \bigcup_{p \text{ Polynom}} TIME(p(n))$$

Definition

Für eine Funktion $f : \mathbb{N} \rightarrow \mathbb{N}$ sei die Klasse $NTIME(f(n))$ genau die Menge der Sprachen L , für die es eine stets anhaltende Mehrband-NTM M gibt mit $L(M) = L$ und $ntime_M(w) \leq f(|w|)$ für alle $w \in \Sigma^*$.

Definition

Die Klasse \mathcal{NP} ist definiert als

$$\mathcal{NP} := \bigcup_{p \text{ Polynom}} NTIME(p(n))$$

Wiederholung: \mathcal{P} vs. \mathcal{NP}

Die Frage „Gilt $\mathcal{P} = \mathcal{NP}$ oder $\mathcal{P} \neq \mathcal{NP}$?“ ist bis heute **ungelöst**.

- ▶ $\mathcal{P} \subseteq \mathcal{NP}$ ist klar.
- ▶ Es gibt gute Gründe, $\mathcal{P} \neq \mathcal{NP}$ zu vermuten.

Bedeutung von \mathcal{P} vs. \mathcal{NP}

Obwohl man die P-vs.-NP-Frage nicht geklärt hat,
will man wissen, wie schwer ein Problem ist:

Bedeutung von \mathcal{P} vs. \mathcal{NP}

Obwohl man die P-vs.-NP-Frage nicht geklärt hat, will man wissen, wie schwer ein Problem ist:

- ▶ Wenn man weiß, dass das Problem in \mathcal{P} liegt, dann existiert ein **effizienter Algorithmus**.

Bedeutung von \mathcal{P} vs. \mathcal{NP}

Obwohl man die P-vs.-NP-Frage nicht geklärt hat, will man wissen, wie schwer ein Problem ist:

- ▶ Wenn man weiß, dass das Problem in \mathcal{P} liegt, dann existiert ein **effizienter Algorithmus**.
- ▶ Wenn man nur weiß, dass das Problem in \mathcal{NP} liegt, dann kennt man nur Algorithmen, die in **deterministischer Exponentialzeit laufen**.

Bedeutung von \mathcal{P} vs. \mathcal{NP}

Obwohl man die P-vs.-NP-Frage nicht geklärt hat, will man wissen, wie schwer ein Problem ist:

- ▶ Wenn man weiß, dass das Problem in \mathcal{P} liegt, dann existiert ein **effizienter Algorithmus**.
- ▶ Wenn man nur weiß, dass das Problem in \mathcal{NP} liegt, dann kennt man nur Algorithmen, die in **deterministischer Exponentialzeit laufen**.

Heute: **\mathcal{NP} -Vollständigkeit**:

Zeige, dass ein gegebenes Problem zu den **schwersten Problemen** in \mathcal{NP} zählt.

Definition

Seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.

Dann sagen wir L_1 ist auf L_2 **polynomiell reduzierbar** (geschrieben $L_1 \leq_p L_2$), falls es eine **totale und in deterministischer Polynomialzeit berechenbare** Funktion

$f : \Sigma_1^* \rightarrow \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1$ g.d.w. $f(w) \in L_2$.

Die Funktion f nennt man **Polynomialzeit-Reduktion**.

Definition

Seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.

Dann sagen wir L_1 ist auf L_2 **polynomiell reduzierbar** (geschrieben $L_1 \leq_p L_2$), falls es eine **totale und in deterministischer Polynomialzeit berechenbare** Funktion $f : \Sigma_1^* \rightarrow \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1$ g.d.w. $f(w) \in L_2$.

Die Funktion f nennt man **Polynomialzeit-Reduktion**.

Die Definition von $L_1 \leq_p L_2$ ist analog zu der von $L_1 \leq L_2$, mit dem Zusatz, dass f in deterministischer Polynomialzeit berechenbar sein muss.

Definition

Seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.

Dann sagen wir L_1 ist auf L_2 **polynomiell reduzierbar** (geschrieben $L_1 \leq_p L_2$), falls es eine **totale und in deterministischer Polynomialzeit berechenbare** Funktion $f : \Sigma_1^* \rightarrow \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1$ g.d.w. $f(w) \in L_2$.

Die Funktion f nennt man **Polynomialzeit-Reduktion**.

Die Definition von $L_1 \leq_p L_2$ ist analog zu der von $L_1 \leq L_2$, mit dem Zusatz, dass f in deterministischer Polynomialzeit berechenbar sein muss.

Nächste Analogie:

$$\begin{aligned} L_1 \leq L_2 \text{ und } L_2 \text{ (semi-)entscheidbar} \\ \implies L_1 \text{ (semi-)entscheidbar} \end{aligned}$$

$$\begin{aligned} L_1 \leq_p L_2 \text{ und } L_2 \in (\mathcal{N})\mathcal{P} \\ \implies L_1 \in (\mathcal{N})\mathcal{P} \end{aligned}$$

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{P}$ und M_2 eine DTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in deterministischer Polynomialzeit anhält.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{P}$ und M_2 eine DTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in deterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f und M_2 . Dann gilt $L(M_f; M_2) = L_1$.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{P}$ und M_2 eine DTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in deterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f und M_2 . Dann gilt $L(M_f; M_2) = L_1$.

$M_f; M_2$ hält stets in deterministischer Polynomialzeit.

Nachweis der Zugehörigkeit zu \mathcal{P}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{P}$, dann gilt $L_1 \in \mathcal{P}$.

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{P}$ und M_2 eine DTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in deterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f und M_2 . Dann gilt $L(M_f; M_2) = L_1$.

$M_f; M_2$ hält stets in deterministischer Polynomialzeit.

Daher gilt $L_1 \in \mathcal{P}$. □

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{NP}$ und M_2 eine NTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in nichtdeterministischer Polynomialzeit anhält.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{NP}$ und M_2 eine NTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in nichtdeterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f (deterministisch) und M_2 (nichtdeterministisch). Dann gilt: $L(M_f; M_2) = L_1$.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{NP}$ und M_2 eine NTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in nichtdeterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f (deterministisch) und
 M_2 (nichtdeterministisch). Dann gilt: $L(M_f; M_2) = L_1$.

$M_f; M_2$ hält stets in nichtdeterministischer Polynomialzeit.

Nachweis der Zugehörigkeit zu \mathcal{NP}

Lemma

Falls $L_1 \leq_p L_2$ und $L_2 \in \mathcal{NP}$, dann gilt $L_1 \in \mathcal{NP}$.

Der Beweis ist analog:

Beweis Seien $L_1 \leq_p L_2$ und f in Polynomialzeit berechenbar.

Sei M_f die DTM, die f in Polynomialzeit berechnet.

Seien $L_2 \in \mathcal{NP}$ und M_2 eine NTM, sodass $L(M_2) = L_2$,
wobei M_2 stets in nichtdeterministischer Polynomialzeit anhält.

Sei $M_f; M_2$ die Hintereinanderausführung von M_f (deterministisch) und
 M_2 (nichtdeterministisch). Dann gilt: $L(M_f; M_2) = L_1$.

$M_f; M_2$ hält stets in nichtdeterministischer Polynomialzeit.

Daher gilt $L_1 \in \mathcal{NP}$. □

Transitivität der Polynomialzeit-Reduktion

Lemma

Die Relation \leq_p ist transitiv, d.h. wenn $L_1 \leq_p L_2$ und $L_2 \leq_p L_3$, dann gilt auch $L_1 \leq_p L_3$.

Transitivität der Polynomialzeit-Reduktion

Lemma

Die Relation \leq_p ist transitiv, d.h. wenn $L_1 \leq_p L_2$ und $L_2 \leq_p L_3$, dann gilt auch $L_1 \leq_p L_3$.

Beweis Die Komposition von zwei Polynomen bleibt ein Polynom. □

Definition

Eine Sprache L heißt \mathcal{NP} -vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. L ist \mathcal{NP} -schwer: für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$.

Definition

Eine Sprache L heißt \mathcal{NP} -vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. L ist \mathcal{NP} -schwer: für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$.

\mathcal{NP} -vollständige Probleme sind die schwierigsten Probleme in \mathcal{NP} .

Definition

Eine Sprache L heißt \mathcal{NP} -vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. L ist \mathcal{NP} -schwer: für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$.

\mathcal{NP} -vollständige Probleme sind die schwierigsten Probleme in \mathcal{NP} .

\mathcal{NP} -Schwere besagt, dass man mit dem \mathcal{NP} -vollständigen Problem **alle anderen** Probleme aus \mathcal{NP} lösen kann.

Definition

Eine Sprache L heißt \mathcal{NP} -vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. L ist \mathcal{NP} -schwer: für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$.

\mathcal{NP} -vollständige Probleme sind die schwierigsten Probleme in \mathcal{NP} .

\mathcal{NP} -Schwere besagt, dass man mit dem \mathcal{NP} -vollständigen Problem **alle anderen** Probleme aus \mathcal{NP} lösen kann.

\mathcal{NP} -schwer wird manchmal auch \mathcal{NP} -hart genannt.

Nachweis der \mathcal{NP} -Vollständigkeit

Nachweis der \mathcal{NP} -Vollständigkeit einer Sprache L :

1. Zugehörigkeit zu \mathcal{NP} :

Gib eine Polynomialzeit-beschränkte NTM an, die L entscheidet.

(Alternativ: Gib eine Polynomialzeit-Reduktion von $L \leq_p L_1$ an mit $L_1 \in \mathcal{NP}$.)

Nachweis der \mathcal{NP} -Vollständigkeit

Nachweis der \mathcal{NP} -Vollständigkeit einer Sprache L :

1. Zugehörigkeit zu \mathcal{NP} :

Gib eine Polynomialzeit-beschränkte NTM an, die L entscheidet.

(Alternativ: Gib eine Polynomialzeit-Reduktion von $L \leq_p L_1$ an mit $L_1 \in \mathcal{NP}$.)

2. \mathcal{NP} -Schwere:

Statt jedes mal neu zu beweisen, dass **alle** Probleme aus \mathcal{NP} auf L polynomiell reduzierbar sind, wähle ein \mathcal{NP} -schweres Problem L_0 und zeige $L_0 \leq_p L$.

Nachweis der \mathcal{NP} -Vollständigkeit

Nachweis der \mathcal{NP} -Vollständigkeit einer Sprache L :

1. Zugehörigkeit zu \mathcal{NP} :

Gib eine Polynomialzeit-beschränkte NTM an, die L entscheidet.

(Alternativ: Gib eine Polynomialzeit-Reduktion von $L \leq_p L_1$ an mit $L_1 \in \mathcal{NP}$.)

2. \mathcal{NP} -Schwere:

Statt jedes mal neu zu beweisen, dass **alle** Probleme aus \mathcal{NP} auf L polynomiell reduzierbar sind, wähle ein \mathcal{NP} -schweres Problem L_0 und zeige $L_0 \leq_p L$.

Da L_0 \mathcal{NP} -schwer, gilt $L' \leq_p L_0$ für alle $L' \in \mathcal{NP}$ und damit $L' \leq_p L_0 \leq_p L$ und mit Transitivität: $L' \leq_p L$ für alle $L' \in \mathcal{NP}$.

Nachweis der \mathcal{NP} -Vollständigkeit

Nachweis der \mathcal{NP} -Vollständigkeit einer Sprache L :

1. Zugehörigkeit zu \mathcal{NP} :

Gib eine Polynomialzeit-beschränkte NTM an, die L entscheidet.

(Alternativ: Gib eine Polynomialzeit-Reduktion von $L \leq_p L_1$ an mit $L_1 \in \mathcal{NP}$.)

2. \mathcal{NP} -Schwere:

Statt jedes mal neu zu beweisen, dass **alle** Probleme aus \mathcal{NP} auf L polynomiell reduzierbar sind, wähle ein \mathcal{NP} -schweres Problem L_0 und zeige $L_0 \leq_p L$.

Da L_0 \mathcal{NP} -schwer, gilt $L' \leq_p L_0$ für alle $L' \in \mathcal{NP}$ und damit $L' \leq_p L_0 \leq_p L$ und mit Transitivität: $L' \leq_p L$ für alle $L' \in \mathcal{NP}$.

Daher ist L \mathcal{NP} -schwer.

Nachweis der \mathcal{NP} -Schwere

Analog zum Vorgehen wie bei der Unentscheidbarkeit, wesentlicher Unterschied:
Polynomialzeit-Reduktion:

$L_1 \leq L_2$ und L_1 unentscheidbar
 $\implies L_2$ unentscheidbar

$L_1 \leq_p L_2$ und L_1 \mathcal{NP} -schwer
 $\implies L_2$ \mathcal{NP} -schwer

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Beweis

← Offensichtlich.

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Beweis

\Leftarrow Offensichtlich.

\Rightarrow Sei L \mathcal{NP} -vollständig und $L \in \mathcal{P}$.

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Beweis

\Leftarrow Offensichtlich.

\Rightarrow Sei L \mathcal{NP} -vollständig und $L \in \mathcal{P}$.

Aus \mathcal{NP} -Schwere von L folgt:

Für alle $L' \in \mathcal{NP}$: $L' \leq_p L$ und damit $L' \in \mathcal{P}$.

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Beweis

\Leftarrow Offensichtlich.

\Rightarrow Sei L \mathcal{NP} -vollständig und $L \in \mathcal{P}$.

Aus \mathcal{NP} -Schwere von L folgt:

Für alle $L' \in \mathcal{NP}$: $L' \leq_p L$ und damit $L' \in \mathcal{P}$.

Da dies für alle $L' \in \mathcal{NP}$ gilt, folgt $\mathcal{P} = \mathcal{NP}$. □

Bedingung für $\mathcal{P} = \mathcal{NP}$

Satz

Sei L ein \mathcal{NP} -vollständiges Problem. Dann gilt $L \in \mathcal{P}$ g.d.w. $\mathcal{P} = \mathcal{NP}$.

Beweis

\Leftarrow Offensichtlich.

\Rightarrow Sei L \mathcal{NP} -vollständig und $L \in \mathcal{P}$.

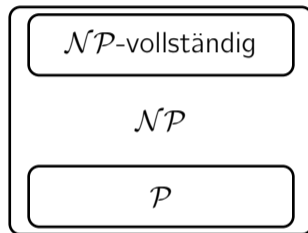
Aus \mathcal{NP} -Schwere von L folgt:

Für alle $L' \in \mathcal{NP}$: $L' \leq_p L$ und damit $L' \in \mathcal{P}$.

Da dies für alle $L' \in \mathcal{NP}$ gilt, folgt $\mathcal{P} = \mathcal{NP}$. □

Also: Es reicht aus nachzuweisen, dass ein \mathcal{NP} -vollständiges Problem in \mathcal{P} bzw. nicht in \mathcal{P} liegt, um die \mathcal{P} -vs.- \mathcal{NP} -Frage ein für allemal beantworten zu können.

Vermutete Lage der Probleme



Unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ gibt es Probleme in \mathcal{NP} gibt, die nicht in \mathcal{P} liegen und nicht \mathcal{NP} -vollständig sind (Ladner 1975).

Was fehlt noch? Ein erstes Problem L_0 , dass man direkt als \mathcal{NP} -vollständig beweist. Ein solches L_0 und den \mathcal{NP} -Vollständigkeitsbeweis sehen wir in der nächsten Vorlesung.

Was fehlt noch? Ein erstes Problem L_0 , dass man direkt als \mathcal{NP} -vollständig beweist. Ein solches L_0 und den \mathcal{NP} -Vollständigkeitsbeweis sehen wir in der nächsten Vorlesung.

Danach können wir \mathcal{NP} -Vollständigkeit von L zeigen durch

1. $L \in \mathcal{NP}$
2. $L_0 \leq_p L$.

Was fehlt noch? Ein erstes Problem L_0 , dass man direkt als \mathcal{NP} -vollständig beweist. Ein solches L_0 und den \mathcal{NP} -Vollständigkeitsbeweis sehen wir in der nächsten Vorlesung.

Danach können wir \mathcal{NP} -Vollständigkeit von L zeigen durch

1. $L \in \mathcal{NP}$
2. $L_0 \leq_p L$.

Danach lernen wir eine Auswahl an \mathcal{NP} -vollständigen Problemen kennen.