Vorlesungsmitschrift

Algorithmen und Berechenbarkeit

Vorlesung 18

Letztes Update: 2018/01/27 - 10:59 Uhr

Die Komplexitätsklassen $\mathcal{NP}$ und $\mathcal{P}$

	$\mathcal{NP}$	$\mathcal{P}$
TM	nichtdeterministisch	deterministisch
Übergang	Relation	Funktion
Akzeptanz	falls es einen gültigen Berechnungspfad zu einem akzeptierenden Endzustand gibt	falls die TM in einem akzeptierenden Endzustand landet (es gibt nur einen Pfad)
Laufzeit für eine Eingabe der Länge n	$t_{\mathcal{M}} =$ *Längster kürzester akzeptierender Berechnungspfad*	$t_{\mathcal{M}} =$ *Längster Berechnungspfad*
Charakterisierung	$\mathcal{NP} \rightarrow$ Entscheidungsprobleme, für die die NTM $\mathcal{M}_{NTM}$ akzeptiert mit Laufzeit $t_{\mathcal{M}} = \mathcal{O}(n^{\alpha})$ **für ein konstantes** α	$\mathcal{P} \rightarrow$ Entscheidungsprobleme, für die die DTM $\mathcal{M}_{DTM}$ akzeptiert mit Laufzeit $t_{\mathcal{M}} = \mathcal{O}(n^{\alpha})$
Beispiele	CLIQUE, Knapsack	Sortieren, Graphzusammenhang

$$\mathcal{NP} \supseteq \mathcal{P}?$$

Die Klasse $\mathcal{NP}$ kann auch wie folgt charakterisiert werden:

Satz: Eine Sprache $\mathcal{L}$ ist genau dann in $\mathcal{NP}$, wenn es einen Polynomzeitalgorithmus V und ein Polynom p gibt mit

$$x \in L \Leftrightarrow \exists y \in \{0,1\}^* \mid |y| \leq p(|x|)$$

und V akzeptiert $y\#x$.

Beweisidee: Falls $\mathcal{L}$ in $\mathcal{NP}$ ist, existiert eine NTM $\mathcal{M}$ mit $\mathcal{L}(\mathcal{M}) = \mathcal{L}$ und polynomieller Laufzeit. Nun modifiziert man $\mathcal{M}$ zu einer deterministischen TM V, die bei jeder (config, gekennzeichnet) - Situation, in der mehrere Schritte möglich sind, den Leitstring y liest und entsprechend deterministisch handelt.
Der Leitstring y muss nur polynomiell lang sein, da die NTM $\mathcal{M}$ polynomielle Laufzeit hat *(weitere Informationen im Skript)*.

Bin-Packing (Optimierung: $\mathcal{NP}$-schwer)

Gegeben: $b \in \mathbb{N}, \quad w_1, w_2, \ldots, w_{\mathbb{N}} \in \{1, \ldots, b\}$

Gegesucht: Eine Funktion $f : \{1, \ldots, N\} \to \{1, \ldots, k\}$, sodass für alle $i \in \{1, \ldots, k\}$ gilt

$$\sum_{j \in f^{-1}(i)} w_j \leq b$$

und k minimal.

Bin-Packing (Entscheidungsvariante: $\mathcal{NP}$-vollständig)

Gegeben: $b \in \mathbb{N}, \quad w_1, w_2, \ldots, w_{\mathbb{N}} \in \{1, \ldots, b\}$

Frage: Existiert eine solche Funktion f?

Travelling Salesperson Problem (TSP)

Gegeben sei ein vollständiger gewichteter Graph mit N-Knoten. Es soll nun eine Permutation π der Knoten gefunden werden, sodass die folgende Gleichung minimal wird:

$$\sum_{i=0}^{n-1} c(\pi(i)), \pi((i+1)n)$$

„Finde die billigste Rundtour“

Satz: Die Entscheidungsvarianten von KP, BPP und TSP sind in $\mathcal{NP}$.

Beweisidee: Man rät eine nichtdeterministische Lösung und verifiziert dann, dass die Lösung in der Tat gültig ist (alles in polynomieller Zeit).

Satz: Wenn die Entscheidungsvariante von KP in polynomieller Zeit lösbar ist, dann auch die Optimierungsvariante.

Beweisidee: Man nimmt an, ein deterministischer-polyzeit-Algorithmus $\mathcal{A}_\epsilon$ für die Entscheidungsvariante existiert.

1. **Schritt**: Man bestimmt den maximal möglichen Rucksackwert durch Binärsuche mittels $\mathcal{A}_\epsilon$ auf die folgende Weise: Man fragt $\mathcal{A}_\epsilon$, ob es einen Rucksack mit Wert $\sum_{i=1}^{N} P_i$ gibt, falls nein, halbiert man, ... Die Anzahl der Iterationen ist

$$\leq \log \sum p_i \leq n$$

bei Eingabelängen $\leq 2^n$.

2. **Schritt**: Man betrachtet den Gegenstand N und entscheidet, ob dieser eingespart werden soll, wie folgt:

$a)$ Man bestimmt den Max-Wert a für die Gegenstände $1, \ldots, N$

$b)$ Man bestimmt den Max-Wert b für die Gegenstände $1, \ldots, N-1$

$c)$ Falls $a = b$, dann wirft man den Gegenstand N weg, sonst packt man den Gegenstand N ein.

$\Rightarrow$ Man wiederholt die Schritte $a) - c)$ für die Gegenstände $1, \ldots, N-1$

Es erfordert $\mathcal{O}(n)$ Aufrufe für das Bestimmen des Max-Werts. Insgesamt ist die Laufzeit polynomiell. □

Satz: Für jedes Entscheidungsproblem $L \in \mathcal{NP}$ gibt es einen deterministischen Algorithmus bzw. eine deterministische TM $\mathcal{M}$, der L entscheidet und dessen Worst-Case-Laufzeit beschränkt ist durch $2^{q(n)}$ für ein Polynom q.

Beweisidee: Man enumeriert alle möglichen Leitstrings für die Verifizierer, das sind exponentiell viele.