Grundlagen der Theoretischen Informatik

Komplexitätstheorie (VI)

20.07.2016

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

DTIME und NTIME / DSPACE und NSPACE

- **DTIME**(T(n)) ist die Klasse der Sprachen, die von $T(n)$-zeitbeschränkten, DTMn akzeptiert werden.
- **NTIME**(T(n)) ist die Klasse der Sprachen, die von $T(n)$-zeitbeschränkten, NTMn akzeptiert werden.

Basismodell: k-DTM $\mathcal{M}$ davon ein spezielles Eingabeband (**offline DTM**).

- **DSPACE**(S(n)) ist die Klasse der Sprachen, die von $S(n)$-speicherbeschränkten, DTMn akzeptiert werden.
- **NSPACE**(S(n)) ist die Klasse der Sprachen, die von $S(n)$-speicherbeschränkten, NTMn akzeptiert werden.

Entscheidbarkeit

Die Funktionen f sind immer sehr einfache Funktionen, insbesondere sind sie alle berechenbar. In dieser Vorlesung betrachten wir nur Potenzen $f(n) = n^i$.

Zeit: Jede Sprache aus **DTIME**(f(n)) ist entscheidbar: Man warte einfach solange wie $f(n)$ angibt. Falls bis dahin kein "Ja" gekommen ist, ist die Antwort eben "Nein".

Speicher: Jede Sprache aus **DSPACE**(f(n)) ist entscheidbar. Denn es gibt nur **endlich viele verschiedene Konfigurationen**. Falls also die DTM nicht terminiert (was passiert), macht man folgendes: man schreibt alle Konfigurationen auf (die komplette Rechnung). Falls die DTM nicht terminiert, muss sie in eine Schleife laufen (eine Konfiguration erreichen, die sie schon einmal hatte). Das lässt sich feststellen.

Antworten

Antworten (informell)

- **DTIME**$(f(n)) \subseteq$ **NTIME**$(f(n))$ und
- **DSPACE**$(f(n)) \subseteq$ **NSPACE**$(f(n))$.

Versucht man aber eine NTM durch eine DTM zu simulieren, braucht man (vermutlich) exponentiell mehr Zeit.

- **NSPACE**$(f(n)) \subseteq$ **DSPACE**$(f^2(n))$.

- **DTIME**$(f(n)) \subseteq$ **DSPACE**$(f(n))$ und
- **NTIME**$(f(n)) \subseteq$ **NSPACE**$(f(n))$.

Aber **DSPACE**$(f(n))$, **NSPACE**$(f(n))$ sind viel größer.

P,NP,PSPACE

Definition [P, NP, PSPACE]

$$\begin{aligned} \mathbf{P} &:= \bigcup_{i \geqslant 1} \mathbf{DTIME}(n^i) \\ \mathbf{NP} &:= \bigcup_{i \geqslant 1} \mathbf{NTIME}(n^i) \\ \mathbf{PSPACE} &:= \bigcup_{i \geqslant 1} \mathbf{DSPACE}(n^i) \end{aligned}$$

Intuitiv

- Probleme in **P** sind effizient lösbar, jene aus **NP** können in exponentieller Zeit gelöst werden.
- **PSPACE** ist eine sehr große Klasse, weit größer als **P** oder **NP** .

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}$$

Reduktion

Definition (Polynomial-Zeit-Reduzibilität)

Seien L_1, L_2 Sprachen.

L_1 ist Polynomial-Zeit reduzibel auf L_2, bezeichnet mit $L_1 \preceq_{\text{pol}} L_2$,

wenn es eine **Polynomial-Zeit beschränkte DTM** gibt,

die für jede Eingabe w eine Ausgabe $f(w)$ erzeugt, so dass

$$w \in L_1 \text{ gdw } f(w) \in L_2$$

Lemma [Polynomial-Zeit-Reduktionen]

1. Sei L_1 Polynomial-Zeit-reduzibel auf L_2 ($L_1 \preceq_{\text{pol}} L_2$). Dann gilt

   Wenn L_2 in **NP** ist dann ist auch L_1 in **NP**

   Wenn L_2 in **P** ist dann ist auch L_1 in **P**

2. Die Komposition zweier Polynomial-Zeit-Reduktionen ist wieder eine Polynomial-Zeit-Reduktion.

NP

Theorem.
Eine Sprache L is in **NP** genau dann wenn es eine Sprache L' in **P** und ein $k \geq 0$ gibt, so das für alle $w \in \Sigma$ gilt:

$$w \in L \text{ gdw. es gibt ein } c : \langle w, c \rangle \in L' \text{ und } |c| < |w|^k.$$

c wird **Zeuge** (*witness* oder Zertifikat/*certificate*) von w in L genannt.

Eine DTM, die die Sprache L' akzeptiert, wird **Prüfer** (*verifier*) von L genannt.

Wichtig:
Ein Entscheidungsproblem ist in **NP** genau dann wenn **jede Ja-Instanz ein kurzes Zertifikat** hat (d.h. seine Länge polynomial in der Länge der Eingabe ist), welche in polynomial-Zeit verifiziert werden kann.

Vollständige und harte Probleme

Definition [NP-vollständig, NP-hart]

- Eine Sprache L heißt **NP-hart (NP-schwer)** wenn
  jede Sprache $L' \in \mathbf{NP}$ polynomial-zeit-reduzibel auf L ist.
- Eine Sprache L heißt **NP-vollständig** wenn sie
  1. in **NP** ist ($L \in \mathbf{NP}$), und
  2. **NP**-hart ist

Vollständige und harte Probleme

Definition [NP-vollständig, NP-hart]

- Eine Sprache L heißt **NP-hart (NP-schwer)** wenn
  jede Sprache $L' \in \mathbf{NP}$ polynomial-zeit-reduzibel auf L ist.
- Eine Sprache L heißt **NP-vollständig** wenn sie
  1. in **NP** ist ($L \in \mathbf{NP}$), und
  2. **NP**-hart ist

Definition [PSPACE-vollständig, PSPACE-hart]

- Eine Sprache L heißt **PSPACE-hart (PSPACE-schwer)** wenn
  jede Sprache $L' \in \mathbf{PSPACE}$ polynomial-zeit-reduzibel auf L ist.
- Eine Sprache L heißt **PSPACE-vollständig** wenn sie
  1. in **PSPACE** ist ($L \in \mathbf{PSPACE}$) und
  2. **PSPACE**-hart ist

Vollständige und harte Probleme

Bemerkenswert

- Wenn gezeigt werden kann, dass auch nur ein einziges **NP**-hartes Problem in **P** liegt, dann ist $\mathbf{P} = \mathbf{NP}$.
- Wenn $\mathbf{P} \neq \mathbf{NP}$ gilt, dann ist kein einziges **NP**-vollständiges Problem in polynomieller Zeit lösbar.

Vollständige und harte Probleme

Bemerkenswert

- Wenn gezeigt werden kann, dass auch nur ein einziges **NP**-hartes Problem in **P** liegt, dann ist $\mathbf{P} = \mathbf{NP}$.
- Wenn $\mathbf{P} \neq \mathbf{NP}$ gilt, dann ist kein einziges **NP**-vollständiges Problem in polynomieller Zeit lösbar.

Eine Million Euro für den, der das „P = NP“-Problem löst!
(Millenium Probleme)

Vollständige und harte Problem

Wie zeigt man NP-Vollständigkeit?

Um zu zeigen, dass eine Sprache L **NP**-vollständig ist:

- Finde bekanntermaßen **NP**-vollständige Sprache L' und
- reduziere sie auf L:

$$L' \preceq L$$

Das genügt, da jede Sprache aus **NP** auf L' reduzierbar ist und wegen $L' \preceq L$ dann auch auf L.

Hierfür häufig verwendet:
SAT-Problem, d.h.

$$L' = L_{\text{sat}} = \{w \mid w \text{ ist eine erfüllbare aussagenlogische Formel}\}$$

Abgschlossenheit der Komplexitätsklassen

P, PSPACE sind abgeschlossen unter Komplement

Alle Komplexitätsklassen, die mittels **deterministischer Turing-Maschinen** definiert sind, sind **abgeschlossen unter Komplement-Bildung**

Denn:
Wenn eine Sprache L dazu gehört, dann auch ihr Komplement
(einfach die alte Maschine ausführen und die Ausgabe invertieren).

Abgschlossenheit der Komplexitätsklassen

Abgeschlossenheit von NP unter Komplement

Frage:

Ist **NP** abgeschlossen unter Komplementbildung?

Abgschlossenheit der Komplexitätsklassen

Abgeschlossenheit von NP unter Komplement

Frage:

Ist **NP** abgeschlossen unter Komplementbildung?

Antwort:

Keiner weiß es!

Die Kompexitätsklasse co-NP

Definition [co-NP]

co-NP ist die Klasse dr Sprachen deren Komplemente in **NP** liegen:

$$\textbf{co-NP} = \{L \mid \overline{L} \in \textbf{NP}\}$$

Beziehungen zwischen Komplexitätsklassen

Die folgenden Beziehungen sind momentan noch unbekannt

1. **P** $=?$ **NP**.
2. **NP** $=?$ **co-NP**.
3. **P** $=?$ **PSPACE**.
4. **NP** $=?$ **PSPACE**.

Beispiele

NP-vollständige Probleme

- Ist ein (un-) gerichteter Graph hamiltonsch? (**Hamiltonian circle**)
- Ist eine logische Formel erfüllbar? (**Satisfiability**)
- Gibt es in einem Graphen eine Clique der Größe k? (**Clique of size** k)
- Ist ein Graph mit drei Farben zu färben? (3-**colorability**)
- Gibt es in einer Menge von ganzen Zahlen eine Teilmenge mit der Gesamtsumme x? (**Subset Sum**)

NP-vollständige Probleme

Definition [Hamilton Circle]

Hamilton-Kreis: Weg entlang der Kanten in einem Graphen, der jeden Knoten genau einmal besucht und wieder zum Ausgangspunkt zurückkehrt.

$L_{\mathbf{Ham_{undir}}}$**:** Die Sprache, die aus allen ungerichteten Graphen besteht, in denen es einen Hamilton-Kreis gibt.

$L_{\mathbf{Ham_{dir}}}$**:** Die Sprache, die aus allen gerichteten Graphen besteht, in denen es einen Hamilton-Kreis gibt.

NP-vollständige Probleme

Definition[Maximale Clique: L_{Clique_k}]

Eine **Clique** in einem Graphen ist ein **vollständiger Teilgraph von** G.

Für $k \in \mathbb{N}$:

L_{Clique_k} Die Sprache, die aus allen ungerichteten Graphen besteht, die eine Clique der Größe k enthalten.

NP-vollständige Probleme

Definition[k-colorability: $L_{\textbf{Color}_{\leq k}}$]

Ein (ungerichteter) Graph heißt **k-färbbar**, falls jeder Knoten mit einer von k Farben so gefärbt werden kann, dass benachbarte Knoten verschiedene Farben haben.

Für $k \in \mathbb{N}$:

$L_{\textbf{Color}_{\leq k}}$ Die Sprache, die aus allen ungerichteten, mit höchstens k Farben färbbaren Graphen besteht.

NP-vollständige Probleme

Definition[SAT, k-CNF, k-DNF]

DNF: Eine Formel ist in **disjunktiver Normalform**, wenn sie von folgender Form ist:

$$(l_{11} \wedge \ldots \wedge l_{1n_1}) \vee \ldots \vee (l_{m1} \wedge \ldots \wedge l_{mn_m})$$

CNF: Eine Formel ist in **konjunktiver Normalform**, wenn sie von folgender Form ist:

$$(l_{11} \vee \ldots \vee l_{1n_1}) \wedge \ldots \wedge (l_{m1} \vee \ldots \vee l_{mn_m})$$

.....

NP-vollständige Probleme

Definition[SAT, *k*-CNF, *k*-DNF] (Fortsetzung)

k*-DNF:** Eine Formel ist in ***k-*DNF* wenn sie in DNF ist und jede ihrer Konjuntionen genau ***k*** Literale hat.

k*-CNF:** Eine Formel ist in ***k-*CNF* wenn sie in CNF ist und jede ihrer Disjunktion genau ***k*** Literale hat.

NP-vollständige Probleme

Theorem [NP-vollständige Probleme]

Die folgenden Probleme liegen in **NP** und sind **NP**-vollständig:

- L_{sat}
- CNF
- k-CNF für $k \geq 3$

NP-vollständige Probleme

Theorem [NP-vollständige Probleme]

Die folgenden Probleme liegen in **NP** und sind **NP**-vollständig:

- L_{sat}
- CNF
- k-CNF für $k \geq 3$

Theorem [Probleme in P]

Die folgenden Probleme liegen in **P**:

- DNF
- k-DNF für alle k
- 2-CNF

NP-vollständige Probleme

Einige Beispiel-Reduktionen

- $L_{\text{CNF-SAT}} \preceq_{\text{pol}} L_{\text{Clique}_{\leq k}}$,
- $L_{\text{Ham}_{\text{dir}}} \preceq_{\text{pol}} L_{\text{Ham}_{\text{undir}}}$,
- $L_{\text{Ham}_{\text{undir}}} \preceq_{\text{pol}} L_{\text{Ham}_{cost \leq k}}$, L_{Clique_k},
- $L_{\text{SAT}} \preceq_{\text{pol}} L_{\text{3-CNF}}$,
- $L_{\text{SAT}} \preceq_{\text{pol}} L_{\text{CNF-SAT}}$,
- $L_{\text{3-CNF}} \preceq_{\text{pol}} L_{\text{Color}_{\leq k}}$.

NP-vollständige Probleme

Beispiel: CNF-SAT $\preceq_{\text{pol}}$ Clique$_{\leq k}$

Gegeben eine Instanz von CNF
(eine Konjunktion von Klauseln $C_1 \wedge C_2 \wedge \ldots \wedge C_k$)

Wir konstruieren daraus einen Graphen:

Knoten: die Paare (x, i), so dass x ein Literal ist, dass in der Klausel C_i vorkommt.

Kanten: Es gibt eine Kante zwischen (x, i) und (y, j) falls:
(1) $i \neq j$, und
(2) x, y sind nicht komplementär.

Es gilt dann:
Die CNF-Formel ist erfüllbar genau dann,
wenn der zugeordnete Graph eine Clique der Größe k hat.

NP-vollständige Probleme

- Ist ein (un-) gerichteter Graph hamiltonsch? (**Hamiltonian circle**)
- Ist eine logische Formel erfüllbar? (**Satisfiability**)
- Gibt es in einem Graphen eine Clique der Größe k? (**Clique of size** k)
- Ist ein Graph mit drei Farben zu färben? (3-**colorability**)
- Gibt es in einer Menge von ganzen Zahlen eine Teilmenge mit der Gesamtsumme x? (**Subset Sum**)