universität
innsbruck

Einführung in die Theoretische Informatik

Martin Avanzini Christian Dalvit Jamie Hochrainer
Georg Moser Johannes Niederhauser Jonas Schöpf

https://tcs-informatik.uibk.ac.at

universität
innsbruck

Zusammenfassung

Satz

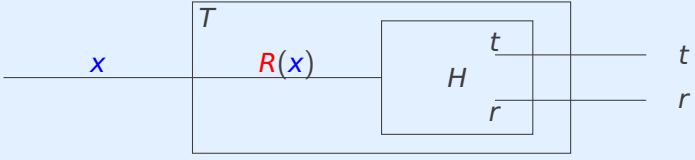
Jede partielle Funktion $f\colon \mathbb{N}^k \to \mathbb{N}$, die berechenbar auf einer RM ist, ist auf einer TM berechenbar und umgekehrt

Definition (Turingreduktion)

angenommen

- L, M Sprachen über Σ
- $L \leqslant_T M$ mit $R\colon \Sigma^* \to \Sigma^*$
- die Reduktion R wird von TM T berechnet, sodass gilt $x \in L \Leftrightarrow R(x) \in M$

Entscheidbarkeit von L, durch Entscheider H von M



Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Formales Beweisen, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

Algebraische Strukturen, Boolesche Algebra, Universelle Algebra

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Chomsky-Hierarchie, Reguläre Sprachen, Kontextfreie Sprachen, Anwendungen von formalen Sprachen

Einführung in die Berechenbarkeitstheorie und Komplexitätstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen, Komplexitätstheorie

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare

universität
innsbruck

Einführung in die Komplexitätstheorie

Definition

Komplexitätstheorie analysiert Algorithmen und Probleme:

Welche Ressourcen benötigt ein bestimmter Algorithmus oder ein Problem?

Ressourcen

- Speicherplatz
- Rechenzeit
- ...

Problem & Algorithmus

Wir unterscheiden zwischen

- der Komplexität eines Algorithmus — Algorithmus von Quine: $2^{c \cdot n}$ (wobei n die maximale binäre Länge der Eingabe)
- der Komplexität eines Problems — SAT ist in NP

Laufzeitkomplexität

Definition

sei M eine totale TM

- die Laufzeitkomplexität von M ist Funktion $T\colon \mathbb{N} \to \mathbb{N}$, wobei T wie folgt definiert

$$T(n) := \max\{m \mid M \text{ hält bei Eingabe } x,\ |x| = n, \text{ nach } m \text{ Schrit-ten}\}$$

- $T(n)$ bezeichnet die Laufzeit von M, wenn n die Länge der Eingabe
- M heißt T-Zeit-Turingmaschine

Definition

sei $T\colon \mathbb{N} \to \mathbb{N}$ eine numerische Funktion

$$\mathrm{DTIME}(T) := \{\mathrm{L}(M) \mid M \text{ ist eine mehrbändige TM mit Laufzeit (ungefähr) in } T\ \}$$

NB: Formal gilt $\exists c \in \mathbb{R}^+\ \exists m\ \forall n \geqslant m$ Laufzeit von M bei Eingabe $x\ \leqslant c \cdot T(n)$, wobei $|x| = n$.

Die Klasse P und NP

Definition

$$\mathsf{P} := \bigcup_{k \geqslant 1} \mathrm{DTIME}(n^k)$$

Beispiel

betrachte SAT als Sprache:

$$\mathrm{SAT} = \{F \mid F \text{ Formel mit erfüllbarer Belegung v}\}$$

es gilt $\mathrm{SAT} \in \mathrm{DTIME}(2^n)$, aber es ist nicht bekannt ob $\mathrm{SAT} \in \mathsf{P}$

Definition

- ein Verifikator einer Sprache $L \subseteq \Sigma^*$, ist ein Algorithmus V sodass
$$L = \{x \in \Sigma^* \mid \exists\, c, \text{ sodass } V \text{ akzeptiert Eingabe } (x, c)\}$$
- ein polytime Verifikator ist ein Verifikator mit (ungefährer) Laufzeit n^k wobei $|x| = n$
- Wort c wird Zertifikat genannt

Definition

NP ist die Klasse der Sprachen, die einen polytime Verifikator haben

Beispiel

- Es gilt SAT $\in$ NP.
- Als Zertifikat wählen wir die (erfüllende) Belegung v für F. Für jede Belegung v kann leicht (in polynomieller Zeit) nachgewiesen werden, ob $\mathrm{v}(F) = \mathrm{T}$.

Reduktionen (in polynomieller Zeit)

Definition

1. $\exists$ k-Band TM M mit Eingabealphabet Σ
2. M läuft in polynomieller Zeit
3. bei Eingabe $x \in \Sigma^*$, schreibt M, $R(x)$ auf das (erste) Band

dann heißt $R\colon \Sigma^* \to \Delta^*$ in polynomieller Zeit berechenbar

Definition

1. $\exists\, R\colon \Sigma^* \to \Delta^*$
2. R berechenbar in polynomieller Zeit
3. für $L \subseteq \Sigma^*$, $M \subseteq \Delta^*$ gilt $x \in L \Leftrightarrow R(x) \in M$

dann ist L in polynomieller Zeit auf M reduzierbar; kurz: $L \leqslant^{\mathrm{p}} M$

Beispiel (Wiederholung)

Seien

$$L = \{x \in \{a,b\}^* \mid |x| \text{ ist gerade}\}$$
$$M = \{x \in \{a,b\}^* \mid x \text{ ist ein Palindrom gerader Länge}\}$$

dann gilt $L \leqslant^{\mathrm{p}} M$

Polynomielle Reduktion

Wir geben eine polynomiell berechenbare Abbildung $R\colon \{a,b\}^* \to \{a,b\}^*$ an, sodass $x \in L \Leftrightarrow R(x) \in M$:

- definiere R', sodass $R'(a) := a$ und $R'(b) := a$
- definiere R als Erweiterung von R' auf Wörter
- R ist eine Stringfunktion, die ein Wort aus $\{a,b\}^n$ in das Wort a^n umwandelt
- Genau dann wenn n gerade ist, ist a^n ein Palindrom gerader Länge

Definition

1. $\mathcal{C}$ eine beliebige Komplexitätsklasse
2. L eine Sprache über Σ
3. $\forall$ Sprachen $M \in \mathcal{C}$ gilt: $M \leqslant^{\mathrm{p}} L$

dann ist L $\leqslant^{\mathrm{p}}$-hart für $\mathcal{C}$

Beispiel

SAT ist $\leqslant^{\mathrm{p}}$-hart für NP, dh. jedes Problem in NP ist auf SAT reduzierbar und außerdem ist SAT $\in$ NP

Definition

für eine Sprache L, sei

1. L $\leqslant^{\mathrm{p}}$-hart für $\mathcal{C}$ und
2. $L \in \mathcal{C}$

dann ist L $\leqslant^{\mathrm{p}}$-vollständig für $\mathcal{C}$ oder (kurz) $\mathcal{C}$-vollständig

universität
innsbruck

Verifikation nach Hoare

Prädikatenlogik (informell)

- Die Prädikatenlogik ist eine Logik, deren Ausdruckskraft über die der Aussagenlogik weit hinausgeht
- Die wichtigste Erweiterung sind Prädikatensymbole und Quantoren
- Prädikatensymbole erlauben es uns, über Elemente einer Menge Aussagen zu treffen

Sprache einer Prädikatenlogik

Eine Prädikatenlogik durch eine Sprache beschrieben, diese Sprache enthält:

1. Funktionssymbole und Prädikatensymbole; Variablen
2. $\underbrace{=}_{\text{Gleichheit}}$, $\underbrace{\neg, \wedge, \vee, \rightarrow}_{\text{Junktoren}}$, $\underbrace{\forall, \exists}_{\text{Quantoren}}$

Beispiel

- Sei 7 eine Konstante und ist_prim ein Prädikatensymbol
- Wir schreiben ist_prim(7), um auszudrücken, dass 7 eine Primzahl

Definition

Ein Ausdruck der mit Hilfe von Variablen und Funktionssymbolen gebildet wird, heißt Term

Definition

1. Sei P ein Prädikatensymbol
2. Seien $t_1, \ldots, t_n$ Terme

Dann nennen wir die Ausdrücke $P(t_1, \ldots, t_n)$ und $t_1 = t_2$ Atome oder atomare Formel

Definition (Zusicherungen)

Wir definieren Zusicherungen induktiv:

1. Atome sind Zusicherungen
2. Wenn A und B Zusicherungen sind, dann sind auch die folgenden Ausdrücke, Zusicherungen:

$$\neg A \quad (A \wedge B) \quad (A \vee B) \quad (A \to B)$$

Konvention

Zusicherungen werden Formeln genannt

Definition

Interpretationen $\mathcal{I}$ werden verwendet, um den Ausdrücken der Prädikatenlogik eine Bedeutung zu geben

Beispiel

- Wir betrachten die Konstante 7 und das Prädikat ist_prim
- Interpretation $\mathcal{I}$ legt fest, dass 7 als die Zahl sieben zu verstehen ist
- $\mathcal{I}$ legt fest, dass das Atom ist_prim(n) genau dann wahr ist, wenn n eine Primzahl

Beobachtung

1. Mit Hilfe von Interpretationen wird der Wahrheitsgehalt von Atomen bestimmt
2. Ist die Wahrheit von Atomen in $\mathcal{I}$ definiert, wird die Wahrheit einer beliebigen Formel durch die Bedeutung der Junktoren bestimmt

Beispiel

Die Formel ist_prim(x) $\wedge$ $x = 7$ bedeutet, dass x die Primzahl 7 ist

Definition

Sei $\mathcal{I}$ eine Interpretation und F eine Formel, wir schreiben $\mathcal{I} \models F$, wenn die Formel F in der Interpretation $\mathcal{I}$ wahr ist

Beispiel

Wenn x die Primzahl 7 ist, dann gilt $\mathcal{I} \models \text{ist\_prim}(x) \wedge x = 7$

Definition

Die Konsequenzrelation $A \models B$ gilt, gdw. für alle Interpretationen $\mathcal{I}$:

$$\mathcal{I} \models A \text{ impliziert } \mathcal{I} \models B$$

Beispiel

Seien $x_1 > 4$ und $x_1 + 1 > 5$ Atome, es gilt:

$$x_1 > 4 \models x_1 + 1 > 5$$

Hoare-Tripel

Definition

- Sei P ein while-Programm (ein Programm einer Registermaschine)
- Seien Q und R Zusicherungen
- Ein Hoare-Tripel ist wie folgt definiert:

$$\{Q\}\ P\ \{R\}$$

- Q wird Vorbedingung
- R wird Nachbedingung genannt

Beispiel

Seien $x_1 > 4$, $x_1 > 5$ Zusicherungen und $x_1 := x_1 + 1$ ein Programm,
dann ist $\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\}$ ein Hoare-Tripel

Definition

- Ein Hoare-Tripel $\{Q\}\ P\ \{R\}$ ist wahr, wenn
  1. Q vor der Ausführung von P gilt
  2. R nach der Ausführung von P gilt
  3. unter der Voraussetzung, dass P terminiert
- Wenn $\{Q\}\ P\ \{R\}$ wahr, dann ist P korrekt in Bezug auf Q und R
- Dann sagen wir auch P ist partiell korrekt
- Das Programm P ist total korrekt, wenn es partiell korrekt ist und terminiert

Beispiel

Die folgenden Hoare-Tripel

$$\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\} \qquad \{x_2 = 0\}\ x_2 := x_2 - 1\ \{x_2 = 0\}$$

sind wahr und die jeweiligen Programme total korrekt

Hoare-Kalkül

Definition

Die Regeln des Hoare-Kalkül sind wie folgt definiert:

$$\text{[z]}\quad \frac{}{\{Q\{x \mapsto t\}\}\ x := t\ \{Q\}} \qquad \text{[a]}\quad \frac{\{Q'\}\ P\ \{R'\}}{\{Q\}\ P\ \{R\}}\ Q \models Q', R' \models R$$

$$\text{[s]}\quad \frac{\{Q\}\ P_1\ \{R\} \quad \{R\}\ P_2\ \{S\}}{\{Q\}\ P_1; P_2\ \{S\}} \qquad \text{[w]}\quad \frac{\{I \wedge B\}\ P\ \{I\}}{\{I\}\ \text{while } B \text{ do } P \text{ end } \{I \wedge \neg B\}}$$

Ist ein Hoare-Tripel in diesem Kalkül ableitbar, dann ist es wahr

Beispiel

$$\frac{\dfrac{}{\{x_1 + 1 > 5\}\ x_1 := x_1 + 1\ \{x_1 > 5\}}\ \text{[z]}}{\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\}}\ \text{[a]}, x_1 > 4 \models x_1 + 1 > 5$$

Beispiel

Wir betrachten das folgende einfache while-Programm P:

$$\begin{array}{l}\text{while } x_i \neq 0 \text{ do}\\ \quad x_i := x_i - 1\\ \text{end}\end{array}$$

und zeigen $\{x_i \geqslant 0\}\ P\ \{x_i = 0\}$

$$\dfrac{\dfrac{\dfrac{}{\{x_i - 1 \geqslant 0\}\ x_i := x_i - 1\ \{x_i \geqslant 0\}}\,[z]}{\{x_i \geqslant 0 \wedge x_i \neq 0\}\ x_i := x_i - 1\ \{x_i \geqslant 0\}}\,[a]}{\{x_i \geqslant 0\}\ P\ \{x_i \geqslant 0 \wedge x_i = 0\}}\,[w]}{\{x_i \geqslant 0\}\ P\ \{x_i = 0\}}\,[a]$$

wir verwenden:

1. $x_i \geqslant 0 \wedge x_i = 0 \models x_i = 0$
2. die Schleifeninvariante $x_i \geqslant 0$
3. $x_i \geqslant 0 \wedge x_i \neq 0 \models x_i - 1 \geqslant 0$