

Diskrete Mathematik

Martin Avanzini Arne Dür Christoph Kollreider Georg Moser

Fakultät für Mathematik, Informatik und Physik @ UIBK
Sommersemester 2010



Zusammenfassung der letzten LV

Problem

MAZE

gegeben

- gerichteter Graph G mit Eckenmenge E
- Knoten $s, t \in E$

∃ Weg zwischen s und t ?

Problem

TSP

gegeben

- n Städte
- Distanz $d_{ij} > 0$, sodass $d_{ij} = d_{ji}$

gesucht: Minimum der Kostensumme:

$$\sum_{i=1}^n d_{\pi(i), \pi(i+1)}$$

$\pi: \{1, \dots, n\} \rightarrow \{1, \dots, n\}$ Permutation mit $\pi(n+1) := \pi(1)$

Problem

das Spiel **verallgemeinerte Länderkunde** ist ein Zweipersonenspiel
Spieler heißen Anna und Otto, gegeben

- gerichteter Graph G und Startknoten s
- Anna beginnt im Land s sie wählt ein erreichbares (unmarkiertes) Land t , welches sie markiert
- Otto zieht von t weiter und markiert wieder
- derjenige Spieler, der nicht mehr ziehen kann verliert

∃ **Gewinnstrategie** für Anna?

Komplexität

- **Platzkomplexität**: $O(n)$

wobei n Größe des Graphen darstellt

Übersicht

Automaten, reguläre Sprachen und Grammatiken, (nicht)-deterministische endliche Automaten, Teilmengenkonstruktion, Automaten mit ϵ -Übergängen, Umwandlung endlicher Automaten in reguläre Ausdrücke, Algebraische Gesetze für reguläre Ausdrücke, Abgeschlossenheit regulärer Sprachen, Pumpinglemma, Minimierung

Einführung in die Berechenbarkeitstheorie, Turing Maschinen, Entscheidungsprobleme, Äquivalente Formulierungen, Universelle Maschinen und Diagonalisierung

Einführung in die Komplexitätstheorie, **Laufzeitkomplexität, die Klassen P und NP**, logarithmisch platzbeschränkte Reduktionen, Speicherplatzkomplexität

Laufzeitkomplexität

Definition

sei M eine deterministische und totale TM

- die **Laufzeitkomplexität** von M ist Funktion $T: \mathbb{N} \rightarrow \mathbb{N}$,
 T misst maximale Anzahl der Schritten (Konfigurationszüge)
 \forall Eingaben der Länge n
- $T(n)$ is die **Laufzeit** von M
- M ist eine **$T(n)$ -Zeit Turingmaschine**

Definition

DTIME($T(n)$)

sei $T: \mathbb{N} \rightarrow \mathbb{N}$ eine numerische Funktion; die **deterministische Zeitkomplexitätsklasse** **DTIME($T(n)$)** ist definiert als:

$$\text{DTIME}(T(n)) = \{L(M) \mid M \text{ ist eine deterministische Mehrband-TM mit Laufzeit in } O(T(n)) \}$$

Die Klassen P und NP

Beispiel

betrachte **MAZE** as Sprache:

$$\text{MAZE} = \{(G, s, t) \mid G \text{ Graph mit Weg von } s \text{ nach } t\}$$

es gilt $\text{MAZE} \in \text{DTIME}(n^2)$

Definition

P

$$P = \bigcup_{k \geq 1} \text{DTIME}(n^k)$$

Definition

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

Definition

NP

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

Beispiel

betrachte TSP als Sprache

$$\text{TSP} = \{(G, b, B) \mid \text{bewerteter Graph } G \text{ hat Hamiltonschen Kreis mit Bewertung } \leq B\}$$

TSP \in NP

definieren einen Verifikator V für TSP mit Eingabe:

- (G, b, B)
- Kreis H

Instanz
Zertifikat

auf der Eingabe:

- prüfe ob H ein Hamiltonscher Kreis ist und Bewertung $\leq B$
 - wenn ja, akzeptiere, sonst verwerfe
- V läuft in polynomieller Zeit

Definition

sei N eine nichtdeterministische und totale TM

- die **Laufzeitkomplexität** von N ist Funktion $T: \mathbb{N} \rightarrow \mathbb{N}$
 T misst die maximale Anzahl von Schritten von N
 \forall Eingaben der Länge n ; auf jedem möglichen Berechnungspfad

Definition

NTIME($T(n)$)

sei $T: \mathbb{N} \rightarrow \mathbb{N}$ eine numerische Funktion; die **nichtdeterministische Zeitkomplexitätsklasse** NTIME($T(n)$) ist definiert als:

$$\text{NTIME}(T(n)) = \{L(N) \mid N \text{ ist eine nichtdeterministische Mehrband-TM mit Laufzeit in } O(T(n))\}$$

Definition

NP'

$$\text{NP}' = \bigcup_{k \geq 1} \text{NTIME}(n^k)$$

Beispiel

TSP \in NP'

Lemma

wenn $L \in \text{NP}'$, dann $L \in \text{NP}$

Beweis

- sei $L \in \text{NP}'$
- \exists NTM N , sodass $L = L(N)$
- N hat polynomielle nichtdeterministische Zeitkomplexität
- definiere einen PolynomZeit Verifikator V mit Eingabe (x, c)
 - 1 simuliere N auf x
 - 2 verwende Zertifikat um Nichtdeterminismus aufzulösen

Algorithmus ist deterministisch
- wenn N akzeptiert, akzeptiert V , ansonsten verwerfe.



Lemma

wenn $L \in \text{NP}$, dann $L \in \text{NP}'$

Beweis

- sei V ein Verifikator für L
- V läuft in polynomieller Zeit, genauer in n^k
- definiere NTM N mit Eingabe x
 - 1 rate (nichtdeterministisch) String c mit $\ell(c) = n^k$
 - 2 simuliere V auf (x, c)
 - 3 wenn V akzeptiert, dann akzeptiert N , sonst verwerfe

Algorithmus ist nichtdeterministisch



Satz

es gilt $\text{NP} = \text{NP}'$: die Klasse der Sprachen, die einen polytime Verifikator haben, ist gleich der Klasse von Sprachen, die durch eine NTM in polynomieller Zeit entschieden werden

Reduktionen in polynomieller Zeit

Definition

wenn

- 1 \exists deterministische, totale TM T mit Eingabealphabet Σ und Bandalphabet Σ
- 2 T läuft in polynomieller Zeit
- 3 bei Eingabe $x \in \Sigma^*$, schreibt T $f(x)$ auf dem (ersten) Band

dann heißt $f: \Sigma^* \rightarrow \Sigma^*$ **in polynomieller Zeit berechenbar**

Definition

$$L \leq_m^p M$$

wenn

- 1 $\exists R: \Sigma^* \rightarrow \Sigma^*$
- 2 R berechenbar in polynomieller Zeit
- 3 für $L \subseteq \Sigma^*$, $M \subseteq \Sigma^*$ gilt $x \in L \iff R(x) \in M$

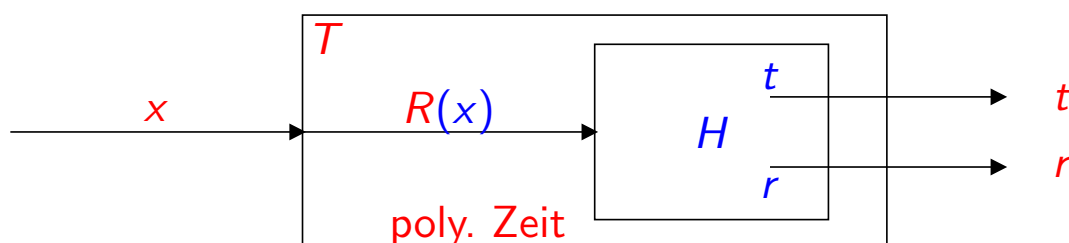
dann ist L auf M **in polynomieller Zeit reduzierbar**

Reduktionen im Bild

Angenommen

- L, M Sprachen über Σ
- $L \leq_m^p M$ mit $R: \Sigma^* \rightarrow \Sigma^*$
- die polynomielle Reduktion R wird von TM T berechnet

$$x \in L \iff R(x) \in M$$



Lemma

wenn $A \leq_m^p B$ und $B \in P$ ($B \in NP$), dann $A \in P$ ($A \in NP$)

Definition

wenn

- 1 \mathcal{C} eine beliebige Komplexitätsklasse
- 2 A eine Sprache über Σ und
- 3 \forall Sprachen $B \in \mathcal{C}$ gilt: $B \leq_m^P A$

dann ist A \leq_m^P -hart für \mathcal{C}

Beispiel

TSP ist \leq_m^P -hart für NP

Definition

wenn

- 1 $A \leq_m^P$ -hart für \mathcal{C} und
- 2 $A \in \mathcal{C}$

dann ist A \leq_m^P -vollständig für \mathcal{C} oder (kurz) \mathcal{C} -vollständig

NP-Vollständigkeit von TSP

Satz

TSP ist \leq_m^P -vollständig für NP

Konsequenz

- wenn \exists polynomieller Algorithmus für TSP, dann
- \exists polynomieller Algorithmus \forall Probleme in NP

Beispiel

MAZE

- MAZE \in NP
- MAZE ist nicht hart für NP

Definition

GG

- GG ist hart für NP
- GG \notin NP

Speicherplatzkomplexität

Definition

sei M eine deterministische und totale TM

- die **Speicherplatzkomplexität** von M ist Funktion $S: \mathbb{N} \rightarrow \mathbb{N}$,
 S misst die maximale Anzahl von Bandfeldern die M liest
 \forall Eingaben der Länge n
- $S(n)$ ist der **Speicherplatz** von M
- M ist eine **$S(n)$ -Platz Turingmaschine**

Definition

sei M eine nichtdeterministische und totale TM

- die **Speicherplatzkomplexität** von M ist Funktion $S: \mathbb{N} \rightarrow \mathbb{N}$,
 S misst die maximale Anzahl von Bandfeldern die M liest
 \forall Eingaben der Länge n ; auf jedem möglichen Weg

Definition

sei $S: \mathbb{N} \rightarrow \mathbb{N}$ eine numerische Funktion

$$\text{DSPACE}(S(n)) = \{L(M) \mid M \text{ ist eine deterministische Mehrband-TM mit Speicherplatz in } O(S(n))\}$$

$$\text{NSPACE}(S(n)) = \{L(N) \mid N \text{ ist eine nichtdeterministische Mehrband-TM mit Speicherplatz in } O(S(n))\}$$

Definition

$$\text{PSPACE} = \bigcup_{k \geq 1} \text{DSPACE}(n^k) \quad \text{NPSPACE} = \bigcup_{k \geq 1} \text{NSPACE}(n^k)$$

Beispiel

- GG ist \leq_m^P -vollständig für PSPACE

Satz

$$P \subseteq NP \subseteq \text{PSPACE} = \text{NPSPACE}$$