Effiziente Berechenbarkeit

bisher: Frage nach der prinzipiellen Lösbarkeit von algorithmischen Fragestellungen

"Was ist überhaupt berechenbar?"

Rekursionstheorie

jetzt: Frage nach der effizienten Lösbarkeit von algorithmischen Fragestellungen

"Was ist mit vernünftigem Aufwand berechenbar?"

Komplexitätstheorie

Komplexitätstheorie

- Untersuchung der algorithmischen Komplexität von entscheidbaren Problemen (Sprachen)
  - welche Ressourcen werden höchstens bzw. mindestens benötigt, um ein Problem zu lösen? *("obere bzw. untere Schranken")*
    - mögliche Ressourcen:
      - Rechenzeit
      - Speicherplatz

      (in einem *"geeigneten"* Berechnungsmodell)

Effiziente Berechenbarkeit

Frage: Wie effizient ist eine (entscheidbare) algorithmische Problemstellung (= Sprache) L lösbar (= erkennbar)?

Ressource, die zum Erkennen von L benötigt wird =

Rechenzeit einer (1-Band) dtm, die L erkennt

oder

Platzbedarf einer (1-Band) dtm, die L erkennt

Deterministische Turing-Maschine

Def.: Eine deterministische Turing-Maschine M (*dtm*) ist ein 7-Tupel $(Q,\Sigma,\Gamma,\delta,q_0,\underline{b},F)$ mit

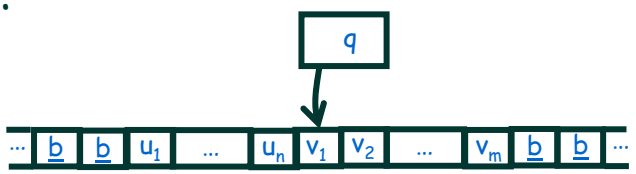
- Q endliche Menge von inneren Zuständen
- Σ endliches Eingabealphabet
- Γ endliches Bandalphabet $\Sigma c \Gamma$
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R,S\}$ Zustandsüberführungsfunktion *(L=links, R=rechts, S=stehenbleiben)*
- $q_0 \in Q$ Anfangszustand
- $\underline{b} \in \Gamma \setminus \Sigma$ Blanksymbol
- $F = F_{ja} \cup F_{nein} c Q$ Menge der Endzustände mit
  - F_{ja} akzeptierenden Endzustände
  - F_{nein} nicht akzeptierenden Endzustände

Konfiguration einer dtm

Def.: Eine Konfiguration einer dtm $M=(Q,\Sigma,\Gamma,\delta,q_0,\underline{b},F)$ ist ein Wort aus $\Gamma^*Q\Gamma^+$

("Momentaufnahme" von M)

Bsp.:



→ Konfiguration $u_1...u_nqv_1..v_m\underline{b}\epsilon\Gamma^*Q\Gamma^+$

Berechnungspfad einer dtm M

Def.:

- $N_M(c)$ bezeichnet Nachfolgekonfiguration von M zur Konfiguration c
- Berechnungspfad von M auf $w\epsilon\Sigma^*$ ist die Folge der Konfigurationen $B_M(w) = c_0,c_1,c_2,...$ die M bei der Abarbeitung von w durchläuft:
  - $c_0 = c(w)$
  - $c_{i+1} = N_M(c_i)$ falls $c_i=uqv$ mit $q\epsilon F^c$

Rechenzeit einer dtm M

Def.: Sei M dtm die auf allen Eingaben hält

- $t_M(w)$:= Rechenzeit von M auf $w \in \Sigma^*$,
  = Länge des Berechnungspfades $B_M(w)$
- $t_M(n) := \max_{w \in \Sigma^n} t_M(w)$ heisst Rechenzeit von M

Platzbedarf einer dtm M

Def.: Sei M dtm die auf allen Eingaben hält

- $s_M(w)$:= Platzbedarf von M auf $w \in \Sigma^*$,
  = maximale Länge einer Konfiguration
  auf dem Berechnungspfad $B_M(w)$
- $s_M(n) := \max_{w \in \Sigma^n} s_M(w)$ heisst Platzbedarf von M

Zeit- und Platzkomplexitätsklassen

Def.: Für $f : \mathbb{N} \to \mathbb{N}$ ist

- $TIME(f(n)) := \{L \subseteq \{0,1\}^* \mid$ es gibt eine dtm M mit $L=L(M)$ und $t_M(n) \leq f(n)\}$
  die von f definierte Zeitkomplexitätsklasse

- $SPACE(f(n)) := \{L \subseteq \{0,1\}^* \mid$ es gibt eine dtm M mit $L=L(M)$ und $s_M(n) \leq f(n)\}$
  die von f definierte Platzkomplexitätsklasse

In polynomieller Zeit erkennbare Sprachen

Def.: $P := \{L \subseteq \{0,1\}^* \mid$ es gibt eine dtm M mit $L=L(M)$ und $t_M(n) \leq n^r$ für ein $r>0\}$

Bsp.: $L = \{0^n1^n \mid n \geq 1\} \in P$

dtm M ändert alternierend eine 0 in ein X und eine 1 in ein Y bis alle 0/1en überschrieben sind → $t_M(n)=O(n^2)$

Robustheit von Komplexitätsklassen

Bem.: $L = \{0^n1^n \mid n \geq 1\}$

2-dtm N kopiert 1^n auf 2. Band und streicht für jede 1 auf Band 2 eine 0 auf Band 1 → $t_N(n)=O(n)$

Frage: *Ändern sich die definierten Komplexitätsklassen, wenn ein anderes Berechnungsmodell zugrunde gelegt wird?*

Zeitkomplexität auf k-dtm

Def. (*temporär*): Für $f : \mathbf{N} \to \mathbf{N}$ ist

- $TIME_k(f(n)) := \{Lc\{0,1\}^* \mid$ es gibt eine k-dtm M mit $L=L(M)$ und $t_M(n) \leq f(n)\}$
- $P_k := \{Lc\{0,1\}^* \mid$ es gibt eine k-dtm M mit $L=L(M)$ und $t_M(n) \leq n^r$ für ein $r>0\}$

Robustheit der Klasse P

Satz: k-dtm M kann von einer 1-dtm S mit

$$t_S(n) = O((t_M(n))^2) \text{ und } s_S(n) = O(s_M(n))$$

simuliert werden.

Folgerung: $P = P_k$

Bew.: Polynome sind unter Komposition abgeschlossen

Bem.: Zahlreiche andere (deterministische) Berechnungsmodelle (z.B. RAM im EKM) können auf 1-dtm mit polynomiellem Zeitverlust simuliert werden

Effiziente Church-Turing These

P ist die Klasse der effizient lösbaren Probleme
(= erkennbaren Sprachen)

Bem.:

- These → *nicht beweisbar*
- gibt keine "Gegenbeispiele"
- bisher wurden keine realistischen stärkeren Maschinenmodelle gefunden (Kandidat: Quantencomputer)

Sprachen in der Klasse P

Bem.: im Folgenden bezeichnet $\langle X \rangle \in \{0,1\}^*$ eine geeignete Kodierung des Objektes X

- RELPRIME = $\{\langle x,y \rangle \mid x,y \in \mathbf{N}$ sind teilerfremd$\} \in \mathbf{P}$
  - Euklidischer Algorithmus
  - ACHTUNG: brute-force funktioniert nicht in polynomieller Zeit !!!
- PRIME = $\{\langle x \rangle \mid x \in \mathbf{N}$ ist Primzahl$\} \in \mathbf{P}$
  - erst seit 2002 bekannt !!!
  - ACHTUNG: brute-force funktioniert nicht in polynomieller Zeit !!!
- DNFSAT = {<F> | F ist erfüllbare aussagenlogische Formel in disjunktiver Normalform} $\in \mathbf{P}$

Sprachen in der Klasse P

- PATH = {<G,s,t> | G ist gerichteter Graph mit gerichtetem Weg von s nach t} $\in \mathbf{P}$
  - BFS/DFS
  - ACHTUNG: brute-force funktioniert nicht in polynomieller Zeit !!!
- EULER = {<G> | G ist ungerichteter Graph, in dem es einen Kreis gibt, der jede Kante genau 1x benutzt} $\in \mathbf{P}$
- SOLVABLE = $\{\langle M \rangle \mid M \in \mathbf{Q}^{k \times k}$ und das LGS $Mx=0$ hat eine Lösung $x \neq 0\} \in \mathbf{P}$
  - Gauss-Elimination

Abschlußeigenschaften der Klasse P

Satz: $L_1, L_2 \in P \rightarrow L_1 \cup L_2, L_1 \cap L_2, L_1^c \in P$

Bew.: *(→ Übung)*