Informatik III

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer
Wintersemester 2006/07
27. Vorlesung
08.02.2007

Komplexitätstheorie - Platzklassen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Platzkomplexität**
  - Definition
  - Simulation mehrerer Bänder
  - Savitchs Theorem
- **PSPACE**
  - PSPACE-schwierig
  - Das quantifizierte Boolesche Erfüllbarkeitsproblem
  - Gewinnstrategien für Spiele
- **Chomsky-Hierachien**
  - Lineare platzbeschränkte TM
  - Das Wortproblem linear platzbeschränkter TMs

Wiederholung: Der Satz von Savitch

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Theorem: Für jede Funktion $s(n) \geq n$**
  - **$NSPACE(s(n)) \subseteq SPACE(s^2(n))$**
- **Beweis:**
  - Betrachte NTM für $L \in NSPACE_1(s(n))$, die mit einer **eindeutigen Konfiguration C_{akz}** akzeptiert
  - Betrachte das Prädikat **`Erreicht-Konf`(C,C',S,T):**
    - Dieses Prädikat ist wahr, wenn die S-Platz-NTM M ausgehend von der Konfiguration C die Konfiguration C' innerhalb von T Schritten erreicht.
  - **Lemma**
    - **`Erreicht-Konf`**(C,C',S,T) kann von einer 2-Band-(S log T)-Platz-DTM entschieden werden
  - **Lemma**
    - Jede s(n)-Platz-NTM hat eine Laufzeit von $2^{O(s(n))}$.
  - **Das Prädikat `Erreicht-Konf`(C_{start}, C_{akz}, s(n), $2^{c\ s(n)}$) entscheidet L.**
  - Dann kann eine 3-Band-DTM L in Platz c s(n) s(n) = c $s(n)^2$ = $O(s^2(n))$ die Sprache L entscheiden

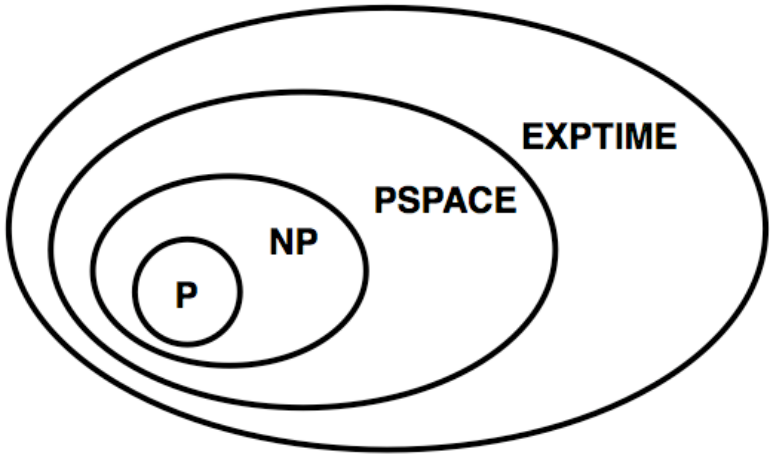
Die Frage P versus NP versus PSPACE

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **P =** Klasse aller Probleme, die effizient *entschieden* werden können
- **NP =** Klasse aller Probleme, die effizient *verifiziert* werden können
- **PSPACE =** Klasse aller Probleme, die auf polynomiellen Platz entschieden werden können
- **EXPTIME =** $\bigcup_k \mathbf{TIME}(2^{n^k})$
- **Man weiß nur, dass P $\neq$ EXPTIME und**

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

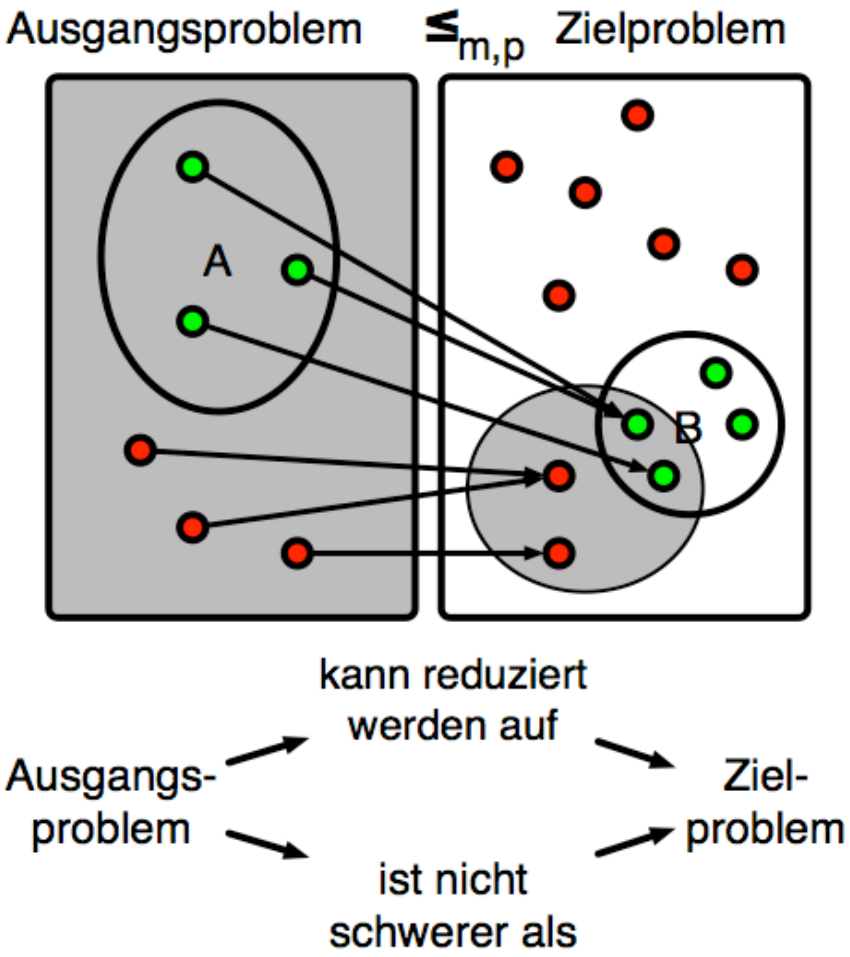
- **Allgemein wird aber vermutet, dass alle Inklusionen echt sind, d.h.**



Die Polynom-Zeit-Abbildungsreduktion

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition (Abbildungsreduktion, Polynomial Time Mapping Reduction, Many-one)**
  - Eine Sprache A kann durch Abbildung auf eine Sprache B in Polynom-Zeit reduziert werden: **$A \leq_{m,p} B$**,
    - falls es eine in Polynom-Zeit berechenbare Funktion $f: \Sigma^* \to \Sigma^*$ gibt,
    - so dass für alle w: $w \in A \Leftrightarrow f(w) \in B$
  - Die Funktion f heißt die **Reduktion** von A auf B.
- **Theorem**
  - Falls $A \leq_{m,p} B$ und B ist in PSPACE, dann ist A auch in PSPACE.

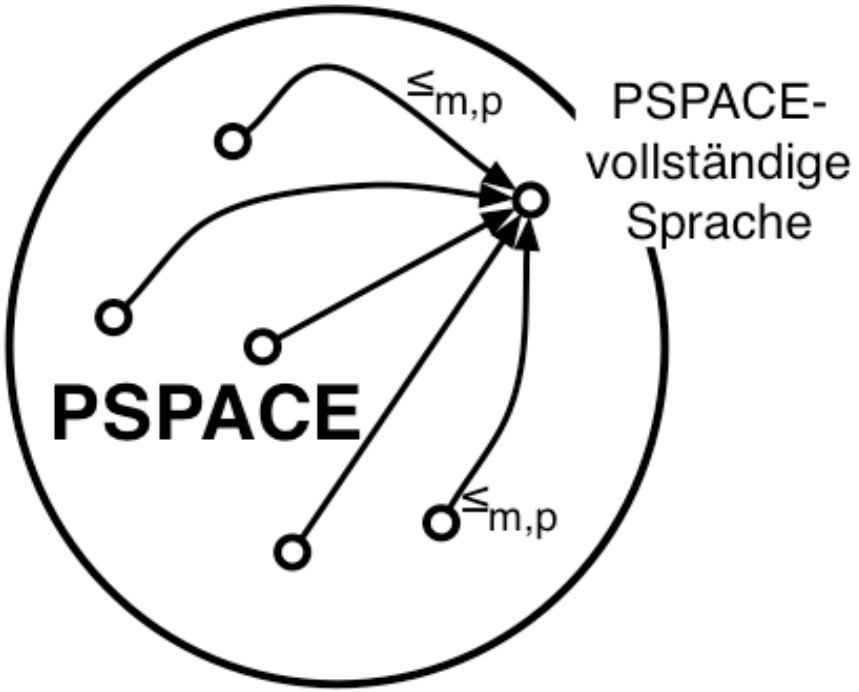


PSPACE-Vollständigkeit

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition:**
  - Eine Sprache S ist **PSPACE-schwierig**, falls
    - für alle L ∈PSPACE: $L \leq_{m,p} S$
  - Eine Sprache S ist **PSPACE-vollständig** wenn:
    - S ∈ PSPACE
    - S ist PSPACE-schwierig

- **Theorem**
  - QBF ist PSPACE-vollständig



Quantifizierte Boolesche Formeln

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Eine Boolesche Funktion ist definiert durch**
  - Eine Konstante 0 oder 1
  - Eine Variable, z.B. x,y,z
  - Die Negation einer Booleschen Funktion, z.B. $\neg$ F(x,y,z)
  - Die Disjunktion zweier Booleschen Funktionen, z.B. F(x,y,z) $\vee$ G(x,y,z)
  - Die Konjunktion zweier Booleschen Funktionen, z.B. F(x,y,z) $\wedge$ G(x,y,z)
- **Eine quantifizierte Boolesche Formel (QBF) besteht aus**
  - Einer Folge von Quantoren $\exists$x, $\forall$y mit daran gebundenen Variablen
  - Einer Booelschen Funktion $F(x_1,x_2,...,x_m)$
  - Jede Variable der Funktion ist genau einmal an einem Quantor gebunden
- **Die quantifizierte Boolesche Formel ist erfüllbar falls**
  - Im Falle eines Existenzquantors: $\exists x\ F(x) \Leftrightarrow F(0) \vee F(1)$
    - wobei F eine weitere QBF sein kann
  - Im Falle eines Allquantors: $\forall x\ F(x) \Leftrightarrow F(0) \wedge F(1)$

Boolesche Funktionen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition QBF (Quantified Boolean Formula Problem)**
  - Das Quantifizierte Boolesche Erfüllbarkeitsproblem der Booleschen Funktion ist definiert als:
  - **QBF = { ϕ | ϕ ist eine wahre quantifizierte Boolesche Formel}**
  - **Gegeben**:
    - Boolesche quantifizierte Formel Q
  - **Gesucht**:
    - Ist Q wahr?

QBF ist in PSPACE

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Theorem**
  - QBF ist in PSPACE
- **Beweis:**
  - Konstruiere TM
    - gegeben QBF: $Q_1x_1\ Q_2x_2\ \ldots\ Q_mx_m\ \phi(x_1,..,x_m)$
      - für $Q_i \in \{\forall,\exists\}$
    - Setze x_1=0: Berechne a= $Q_2x_2\ \ldots\ Q_mx_m\ \phi(x_1,..,x_m)$
    - Setze x_1=1: Berechne b= $Q_2x_2\ \ldots\ Q_mx_m\ \phi(x_1,..,x_m)$
    - Falls $Q_1 = \forall$
      - Falls a und b gib 1 aus, sonst 0.
    - Falls $Q_1 = \exists$
      - Falls a oder b gib 1 aus, sonst 0."
  - Platzbedarf:
    - O(1) in jeder Rekursionstiefe
    - Anzahl Rekursionstiefen: $m \leq n$
    - Gesamtspeicher: O(n)

QBF ist PSPACE-schwierig

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Theorem**
  - Für alle $L \in PSPACE$ gilt $L \leq_{m,p} QBF$
- **Beweis**
  - Betrachte 1-Band s(n)-Platz-TM mit $s(n) = O(n^k)$
  - Dann gibt es eine Boolesche Funktion polynomieller Größe, die wahr ist, falls **`Erreicht-Konf`**(C,C',S,T) für gegebene Eingabelänge und die in Polynom-Zeit beschreibbar ist.
  - **Lemma**
    - **`Erreicht-Konf`**(C,C',S,T) kann von einer quantifizierten Booleschen Funktion der Länge O(S log T) beschrieben werden.
    - Diese Funktion lässt sich von einer DTM in Zeit O(S log T) konstruieren
  - Konstruiere QBF zu **`Erreicht-Konf`**(Anfangskonfiguration,Endkonfiguration,s(n),$2^{cs(n)}$)

Wie man `Erreicht-Konf` nicht darstellt

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Betrachte nun folgende QBF für**
  - **`Erreicht-Konf`**(C,C',S, 0) = ("C=C'")
  - **`Erreicht-Konf`**(C,C',S, 1) = ("C=C'") ∨ ("C geht über in C'")
  - **`Erreicht-Konf`**(C,C',S, 2T) =
    ∃ Z: **`Erreicht-Konf`**(C,Z,S,T) ∧ **`Erreicht-Konf`**(Z,C'S,T)
- **Größe G(T) der Formel für `Erreicht-Konf`:**
  - G(0) = c S
  - G(1) = c' S
  - G(2T) = 2 G(T) + c''
    - für geeignete Konstanten c,c',c' '≥ 1
  - **Beobachtung: G(T) = Ω(S T)**
    - Siehe nächste Folie ...
- **Problem:**
  - Rekursiv definierte Formel wächst linear in T
  - $T = 2^{c\, s(n)}$ notwendig die Berechnung einer s(n)-Platz-DTM
  - Daher ergibt das keine Polynom-Zeit-Reduktion

Die zugehörige Rekursion

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Betrachte Rekursion**
  - $g(0) = S$
  - $g(1) = S$
  - $g(2T) = 2\, g(T)$
- **Beobachtung: $g(t) \leq G(t)$**
  - Beweis durch vollständige Induktion
- **Behauptung: $g(2^k) = S\, 2^k$, für $k \geq 0$**
- **Beweis:**
  - Aussage ist korrekt für $k=0$
  - Angenommen die Aussage ist korrekt für k
  - Dann ist $g(2^{k+1}) = 2\, g(2^k) = 2\, S\, 2^k = S\, 2^{k+1}$
- **Daraus folgt die**
  - **Beobachtung: $G(T) = \Omega(S\, T)$**

Wie man `Erreicht-Konf` darstellt

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Lemma**
  - **`Erreicht-Konf`**(C,C',S,T) kann von einer quantifizierten Booleschen Funktion der Länge O(S log T) beschrieben werden.
  - Diese Funktion lässt sich von einer DTM in Zeit O(S log T) konstruieren Lösung:
- **Beweis: Betrachte nun folgende QBF für**
  - **`Erreicht-Konf`**(C,C',S, 0) = ("C=C'")
  - **`Erreicht-Konf`**(C,C',S, 1) = ("C=C'") ∨ ("C geht über in C'")
  - **`Erreicht-Konf`**(C,C',S, 2T) =
    ∃ Z: ∀ A,B: ("(A,B)=(C,Z)" ∨ "(A,B)=(Z,C')") ⇒ **`Erreicht-Konf`**(A,B,S,T))
  - **Behauptung 1: Die QBF ist korrekt**
    - Voraussetzung die Rekursion:
    - **`Erreicht-Konf`**(C,C',S, 2T) =
      ∃ Z: **`Erreicht-Konf`**(C,Z,S,T) ∧ **`Erreicht-Konf`**(Z,C',S,T)
    - ist korrekt
  - **Behauptung 2: Die Länge der QBF ist O(S log T)**
  - Wahl $T = 2^{c\,S}$ reicht für die Berechnung einer S-Platz-DTM
  - Konstruktion läßt sich in von einer Polynom-Zeit-DTM berechnen, da S log T = $O(S^2)$

Sind die Formeln äquivalent?

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

∃ Z: ∀ A,B:
((A,B)=(C,Z) ∨ (A,B)=(Z,C'))
⇒ `Erreicht-Konf`(A,B,S,T)

Def.: P(A,B)= `Erreicht-Konf`(A,B,S,T)

∀ A,B:
((A,B)=(C,Z) ∨ (A,B)=(Z,C')) ⇒ P(A,B)

∃Z: `Erreicht-Konf`(C,Z,S,T) ∧
`Erreicht-Konf`(Z,C',S,T)

und betrachte nur Term hinter ∃ Z

P(C,Z) ∧ P(Z,C')

$$\bigwedge_{A,B} ((A,B) = C,Z) \vee (A,B) = (Z,C')) \Rightarrow P(A,B)$$

3 Fälle:

- (A,B)=(C,Z): Ergebnis: P(C,Z)
- (A,B)=(Z,C'): Ergebnis: P(Z,C')
- Weder noch: Ergebnis: Wahr

Ergibt:
P(C,Z) ∧ P(Z,C')

Die Länge der QBF ist O(S log T)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Rekursion:**
  - **`Erreicht-Konf`**(C,C',S, 0) = ("C=C'")
  - **`Erreicht-Konf`**(C,C',S, 1) = ("C=C'") ∨ ("C geht über in C'")
  - **`Erreicht-Konf`**(C,C',S, 2T) =
    ∃ Z: ∀ A,B: ("(A,B)=(C,Z)" ∨ "(A,B)=(Z,C')") ⇒ **`Erreicht-Konf`**(A,B,S,T))
- **Sei g(T) die Größe:**
  - g(0) = c S
  - g(1) = c S
  - g(2T) = g(T) + c S
    - für geeignete Konstante c≥1
- **Behauptung: $g(T) = g(2^{\log T}) \leq c\,(1+\log T)\,S$, für $T \geq 0$**
  - Korrekt für T=1
  - Induktionsannahme: Behauptung korrekt für T
  - Induktionsschluss:
    $g(2T) = g(2^{1+\log T})$
    $= g(T) + c\,S$
    $= c\,(1+\log T)\,S + c\,S$
    $= c\,(2+\log T)\,S = c\,(1+\log 2T)\,S$

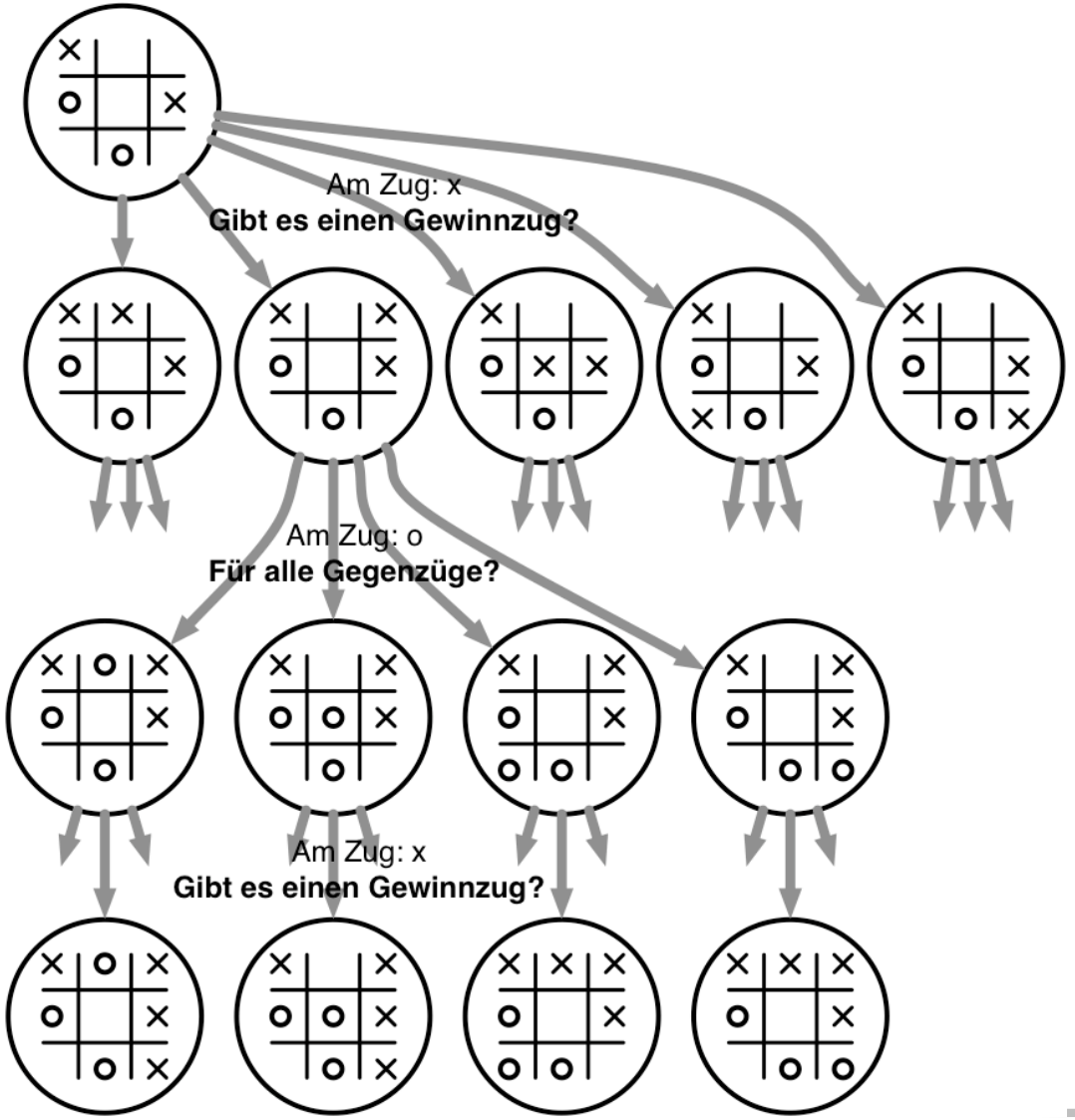
PSPACE und Spiele

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Tic-Tac-Toe kann als quantifizierte Boolesche Funktion beschrieben werden:**
  - Gibt es einen Zug für mich,
  - So dass für alle gültige Züge des Gegners,
  - es einen Gewinnzug für mich gibt oder einen gültigen Zug für mich gibt,
  - so dass für alle gültige Züge des Gegners kein Gewinnzug für ihn ist und
  - es einen Gewinnzug für mich gibt oder einen gültigen Zug für mich gibt,
  - so dass für alle gültige Züge des Gegners kein Gewinnzug für ihn ist und
  - es einen Gewinnzug für mich gibt oder einen gültigen Zug für mich gibt,
  - so dass für alle gültige Züge des Gegners kein Gewinnzug für ihn ist und
  - es einen Gewinnzug für mich gibt.

Tic-Tac-Toe als Spielbaum

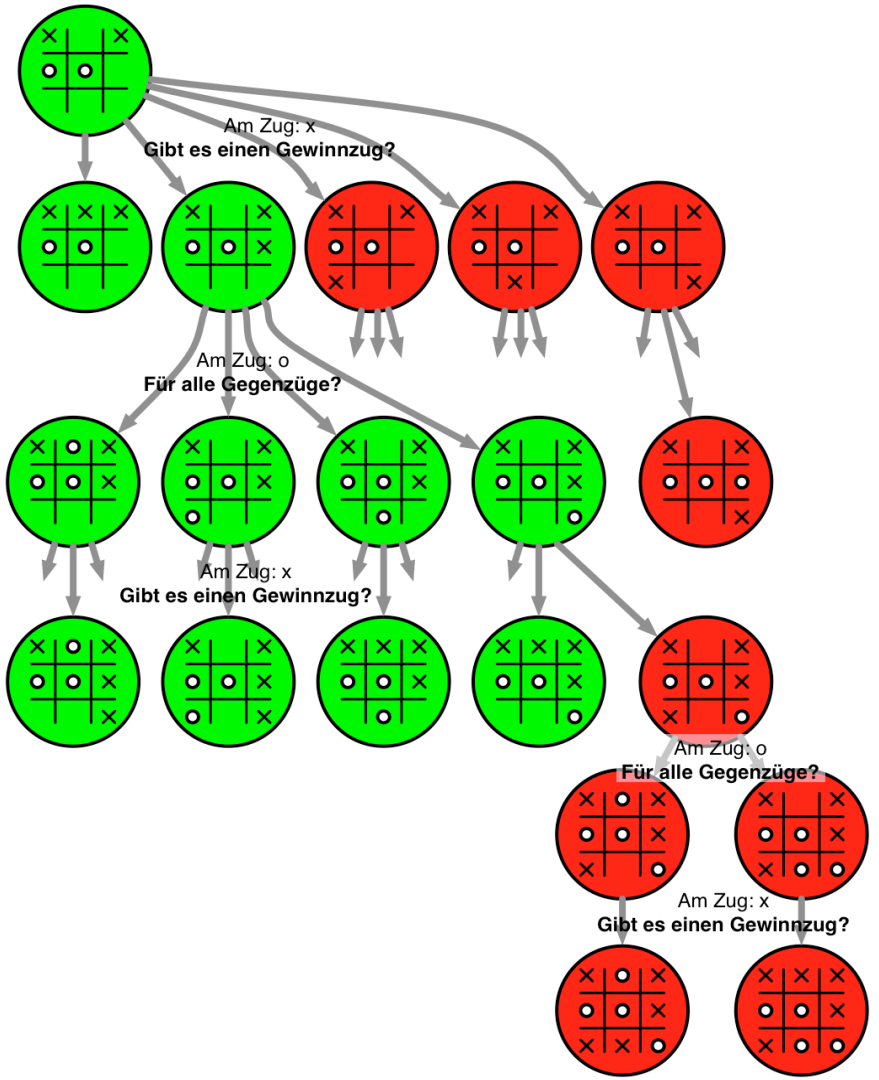
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



Tic-Tac-Toe als Spielbaum

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Eine QBF beantwortet die Frage, ob Spieler x gewinnen kann**
- **Aber welcher Zug ist der Gewinnzug?**
- **Lösung:**
  - Probiere alle Möglichkeiten aus
  - Konstruiere die entsprechende QBF für Spieler o
  - Wähle die Möglichkeit, in der o verliert
- **Diese Art der Reduktion kennen wir als Turing-Reduktion**
- **PSPACE ist abgeschlossen gegenüber Polynom-Zeit-Turing-Reduktionen**

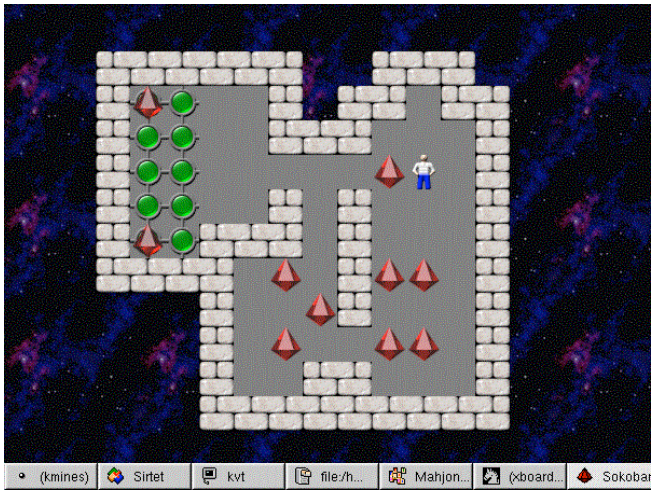


Mehr Spiele

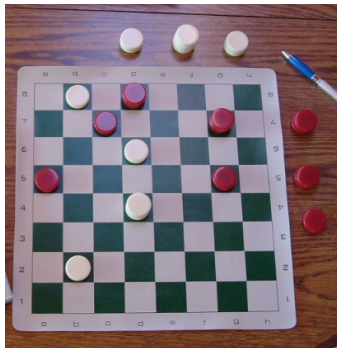
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Schach auf allgemeiner Brettgröße mit Fortschrittsregel ist PSPACE-vollständig**
  - Fortschrittsregel:
    - innerhalb von 50 Zügen muss ein Bauer bewegt werden oder eine Figur geschlagen werden

- **Sokoban:**
  - PSPACE-vollständig



- **Schach in verallgemeinerter Form ohne Fortschrittsregel ist**
  - EXPTIME-vollständig
- **Dame: für allgemeine Brettgrößen**
  - EXPTIME-vollständig



Die Chomsky-Klassifizierung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Chomsky-Hierachien**
  - 3: Reguläre Grammatiken
  - 2: Kontextfreie Grammatiken
  - 1: Wachsende kontextsensitive Grammatiken
  - 0: Allgemeine Grammatiken
- **Alternative Beschreibung**
  - Reguläre Sprachen und konstanter Platz
  - Kontextfreie Sprachen und Kellerautomaten
  - Wachsende kontextsensitive Sprachen und linearer Platz
  - Chomsky-0-Sprachen und Rekursiv aufzählbare Sprachen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Chomsky Typen

Übersicht Chomsky-Charakterisierung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Stufe	Sprache	Regeln	Maschinenmodell
Typ-0	Rekursiv Aufzählbar	keine Einschränkung	Turing-Maschine
Typ-1	(Wachsend) Kontextsensitive Sprachen	$\alpha A\beta \rightarrow \alpha\gamma\beta$ $A \in V$, $\lvert\gamma\rvert > 1$ $\alpha,\beta,\gamma \in (\Sigma \cup V)^*$	Linear-Platz-NTM
Typ-2	Kontextfreie Sprachen	$A \rightarrow \gamma$ $A \in V$ $\gamma \in (\Sigma \cup V)^*$	Nichtdet. Kellerautomat (PDA)
Typ-3	Reguläre Sprachen	$A \rightarrow aB$ $A \rightarrow a$ $A,B \in V$ $a \in \Sigma$	Endlicher Automat (NFA/DFA)

Ende der 27. Vorlesung

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer

Wintersemester 2006/07

27. Vorlesung

08.02.2007