Informatik III

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

Christian Schindelhauer
Wintersemester 2006/07
17. Vorlesung
21.12.2006

Komplexitätstheorie - Zeitklassen

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

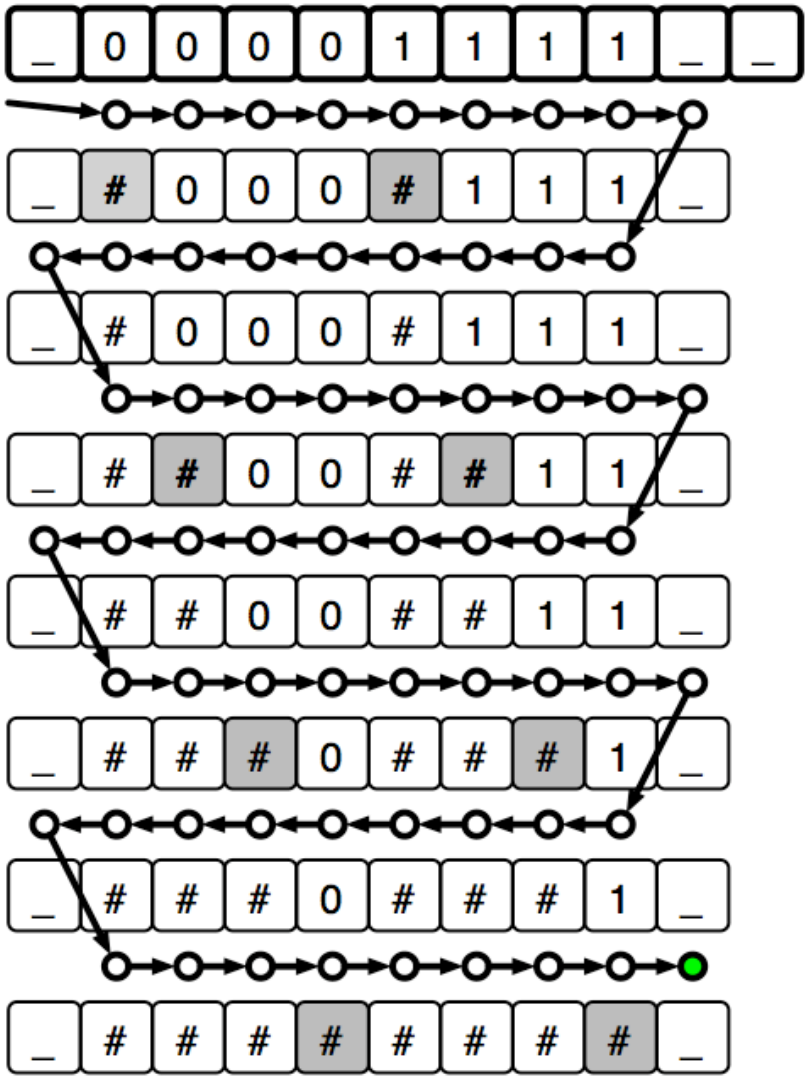
- **Komplexitätsmaße**
  - Wiederholung: O,o,ω,Θ,Ω
  - Laufzeitanalyse
- **Die Komplexitätsklassen TIME,**
  - DTIME, NTIME
  - P mit Beispielen
  - NP mit Beispielen
- **Das Cook-Levin-Theorem**
  - Polynomial-Zeit-Reduktion
  - Reduktionen zwischen 3SAT und Clique
  - NP-vollständigkeit
  - SAT ist NP-vollständig
- **Weitere NP-vollständige Probleme**
  - Knotenüberdeckung (Vertex-Cover)
  - Das Hamiltonsche Pfadproblem
  - Das ungerichtete Hamiltonsche Pfadproblem
  - Das Teilsummenproblem

Laufzeit einer TM - Beispiel

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

- **Beispiel: $A = \{0^k 1^k \mid k \geq 0\}$**

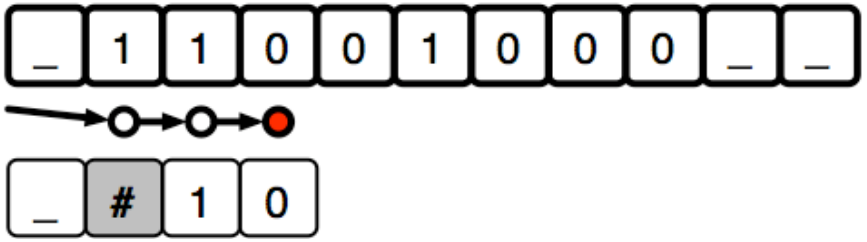
- **Maschine M_1 für A:** "Für Eingabe w:
  1. Laufe von links nach rechts über das Band und verwerfe, falls eine 0 rechts von einer 1 gefunden wird
  2. Wiederhole falls mindestens eine 0 und eine 1 auf dem Band sind:
  3. Laufe auf den Band und lösche eine 0 und eine 1
  4. Falls 0er oder 1er noch bleiben, verwerfe.
     Ansonsten, akzeptiere.



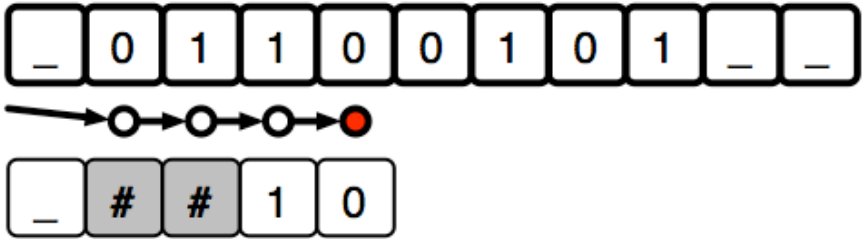
Average versus Worst-Case-Laufzeit

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

- Im **schlimmsten (worst-case)** Laufzeitfall ist das Wort in A
- Dann muss die TM M_1 alle 0er und 1er auskreuzen:
  - Das macht k Runden
- In jeder Runde muss die TM einmal die gesamte Eingabe der Länge n=2k durchlaufen
- Die **Gesamtlaufzeit** ist immer höchstens **$4k^2+2k = n(n-2)$**



- Tatsächlich ist aber die **erwartete Laufzeit** kleiner als 5:
  - Für ein zufälliges Wort ist die Wahrscheinlichkeit, dass eine 0 rechts von einer 1 steht an jeder Stelle 1/2.
- Offensichtlich gibt die **Average-Laufzeit** hier nicht das interessante Verhalten der TM wieder
- Daher betrachtet die Komplexitätstheorie zumeist **nur Worst-Case-Laufzeiten**



Ein Zeit-Komplexitätsmaß

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

- **Definition**
  - Sei M eine deterministische Turing-Maschine, die auf allen Eingaben hält.
  - Die **Laufzeit (Zeitkomplexität)** von M ist eine Funktion f: N→N,
    - wobei f(n) die maximale Anzahl von Schritten von M beschreibt auf einer Eingabe der Länge n.
  - Falls f(n) die Laufzeit einer TM M ist, nennt man M auch eine **f(n)-Zeit-Turing-Maschine**
    - z.B. Linear-Zeit-TM für f(n) = c n für eine Konstante c
    - z.B. Quadrat-Zeit-TM für f(n) = c n^2 für eine Konstante c
    - z.B. Polynom-Zeit-TM für f(n) = c n^k für Konstanten c und k

- Zumeist beschreibt n die Eingabelänge.

Laufzeiten im Vergleich

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

- **Wie kann man Zeitkomplexitäten vergleichen?**
- **Beispiel: 100 n versus n^2 versus $2^n/1000$**
  - n=1 100 1 0,002
  - n=2 200 4 0,004
  - ... ...
  - n=10 1.000 100 1,024
  - ... ...
  - n=20 2.000 400 1049
  - n=21 2.100 481 2098
  - ... ...
  - n=100 10.000 10.000 1.27×10^{27}
  - ... ...
  - n=1.000 100.000 1.000.000 1.07×10^{298}

- **Für große Eingabelängen spielen die konstanten Faktoren eine untergeordnete Rolle**
  - Hardware wird jedes Jahr (noch) um einen konstante Faktor schneller
  - Problemgrößen wachsen auch jedes Jahr

Definition der Asymptotik

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

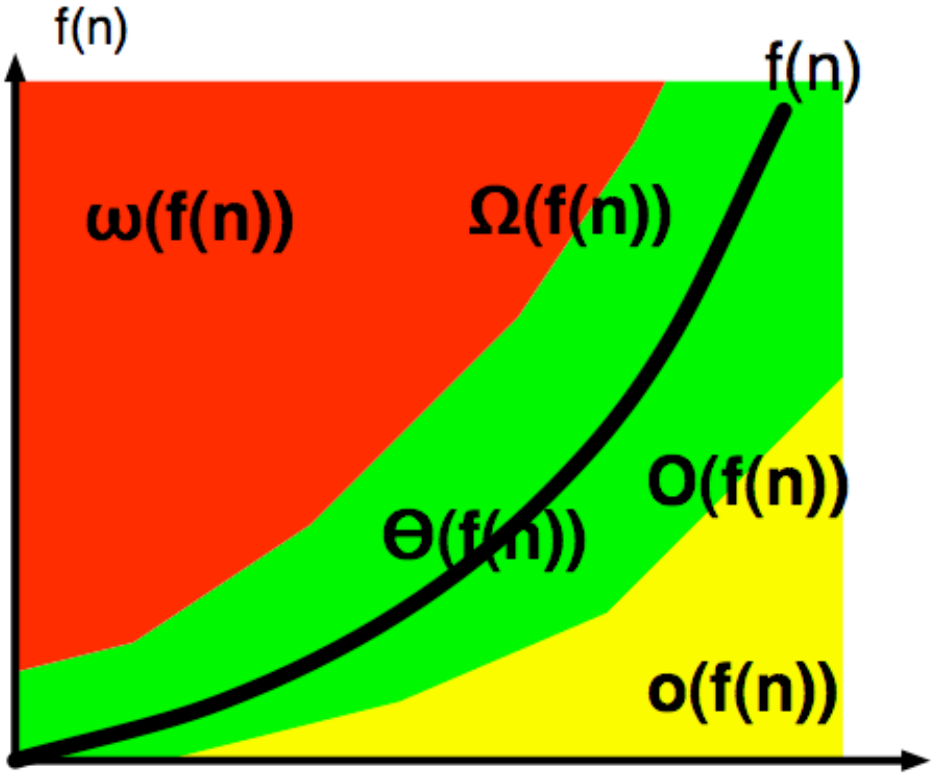
$$f \leq_{ae} g \quad :\Longleftrightarrow \quad \exists n_0 \in \mathbb{R} \quad \forall n \geq n_0 : \quad f(n) \leq g(n) \; .$$

$$\begin{aligned}
O(g) &:= \{f \mid \exists k \in \mathbb{R} \quad f \leq_{ae} k \cdot g\} \; , \\
o(g) &:= \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot f \leq_{ae} g\} \; , \\
\omega(g) &:= \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot g \leq_{ae} f\} \; , \\
\Omega(g) &:= \{f \mid \exists k \in \mathbb{R} \quad g \leq_{ae} k \cdot f\} \; , \\
\Theta(g) &:= O(g) \quad \cap \quad \Omega(g) \; .
\end{aligned}$$

Die asymptotischen Wachstumsklassen

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

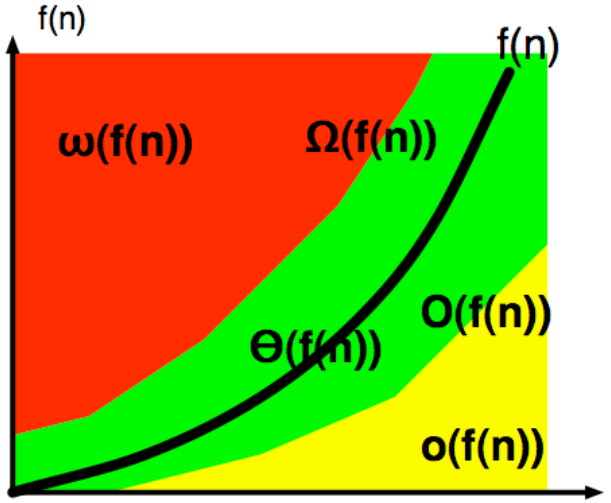
$$
\begin{aligned}
O(g) &:= \{f \mid \exists k \in \mathbb{R} \quad f \leq_{ae} k \cdot g\}, \\
o(g) &:= \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot f \leq_{ae} g\}, \\
\omega(g) &:= \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot g \leq_{ae} f\}, \\
\Omega(g) &:= \{f \mid \exists k \in \mathbb{R} \quad g \leq_{ae} k \cdot f\}, \\
\Theta(g) &:= O(g) \quad \cap \quad \Omega(g).
\end{aligned}
$$



Techniken und Tricks

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

- **Wachstumsklassen erleichtern die Analyse.**
- **Techniken:**
  - Falls die Funktion ein Polynom $a_0+a_1n+a_2n^2+...+a_kn^k$ ist und $a_k>0$, ist die Wachstumsklasse: $\Theta(n^k)$
  - Falls $f \in O(g)$, dann ist $f+g \in O(g)$.
  - $f \in o(g)$ genau dann wenn $g \in \omega(f)$
  - $f \in O(g)$ genau dann wenn $g \in \Omega(f)$
- **Eselsbrücken:**
  - o,O bedeutet: “kleiner”, “kleiner gleich”
  - ω, Ω bedeutet: “größer”, “größer gleich”
  - Θ bedeutet O und Ω - alles Großbuchstaben
    - also “asymptotisch gleich”
- **Theorem:**



$$f \in o(g) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

$$f \in O(g) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} \leq c \text{ für eine Konstante } c$$

Korrekte und “schlampige” Notationen

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

- **Elemente der Wachstumsklassen sind Funktionen:**

$$n \mapsto f(n)$$

- **Diese werden so abgekürzt:**

$$\mathcal{N} := n \mapsto n$$

$$\mathcal{N}^2 := n \mapsto n^2$$

$$2^{\mathcal{N}} := n \mapsto 2^n$$

- **Eine Funktion ist Element einer Wachstumsklasse**

$$\mathcal{N}^2 \in o(2^{\mathcal{N}})$$

$$(n \mapsto n^2) \in o(n \mapsto 2^n)$$

- **Meistens schreibt man einfach nur f(n) und nimmt implizit an, dass n das Argument ist**

$$f(n)$$

- **Man schreibt einfach nur**

$$n$$

$$n^2$$

$$2^n$$

- **Statt dem Element-Zeichen wird “=“ geschrieben:**

$$n^2 = o(2^n)$$

Beispiele:

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

1. $\sqrt{n} = o(n)$
2. $n = o(n \log \log n)$
3. $n \log \log n = o(n \log n)$
4. $n \log n = o(n^2)$
5. $n^2 = o(n^3)$

Zeitkomplexitätsklassen

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

- **Definition**
  - Sei $t:N \rightarrow R^+$ eine Funktion.
  - Die Zeitkomplexitätsklasse **TIME(t(n))** ist die Menge aller Sprachen,
    - die von einer deterministischen O(t(n))-Zeit-Turing-Maschine entschieden werden.
  - Wird die Anzahl der Bänder auf k beschränkt, schreiben wir **$\mathbf{TIME_{k\text{-}Band}(t(n))}$** oder einfach **$\mathbf{TIME_k(t(n))}$.**
- **Beispiel:**
  - $A = \{0^k 1^k \mid k \geq 0\}$
  - Wir haben gesehen: $A \in TIME_1(n^2)$,
    - da es eine TM mit Laufzeit $O(n^2)$ gibt.
- **Frage:**
  - Gibt es eine TM, die A in Linear-Zeit löst?

Eine erste Fragestellung, eine Antwort, noch eine Frage...

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

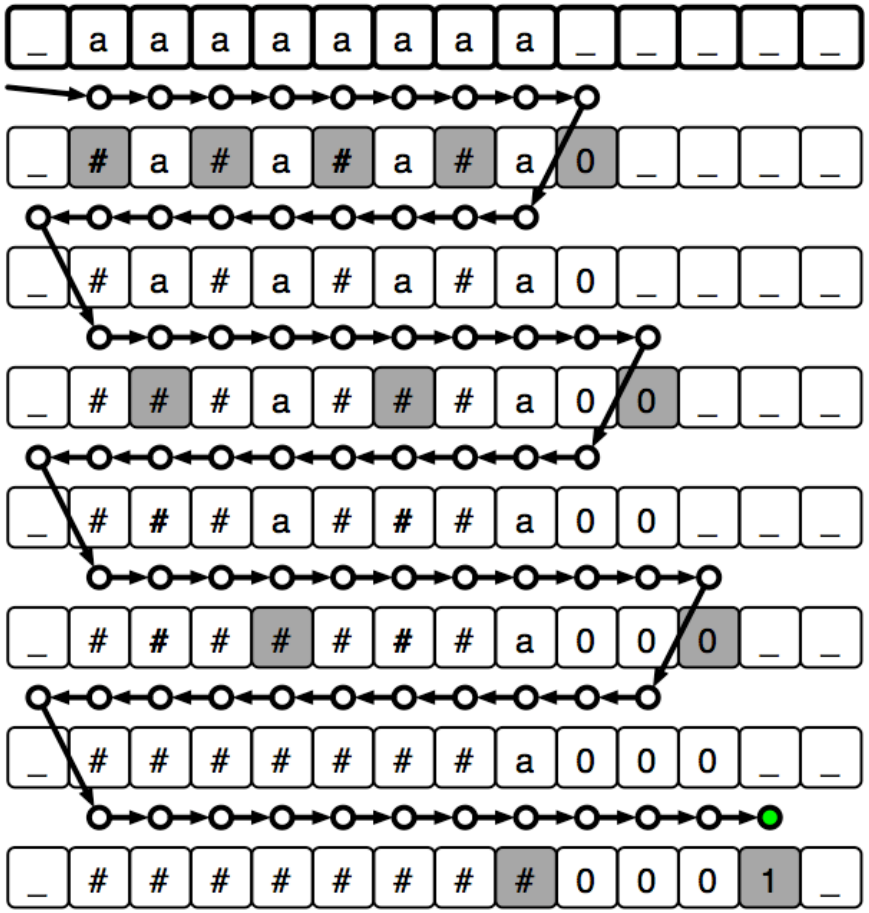
- **Beispiel:**
  - $A = \{0^k1^k \mid k \geq 0\}$
  - $A \in TIME_1(n^2)$,
- **Frage:**
  - Gibt es eine TM, die A in **Linear-Zeit** löst?
- **Antwort:**
  - Ja:
  - Verwende eine TM mit 2 Bändern,
  - Kopiere alle 0er auf das zweite Band und vergleiche dann alle 1er auf dem Eingabeband mit dem zweiten Band.
- **Also: $A \in TIME_2(n)$,**
- **Frage: $A \in TIME_1(o(n^2))$?**
  - D.h. kann man A auch mit nur 1 Band in Zeit $o(n^2)$ lösen?
- **Antwort:**
  - Ja. Es gilt $A \in TIME_{1\text{-Band}}(n \log n)$.
- **Aber wie?**

Idee: Zählen

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

- **Gesucht: TM**
  - mit Eingabe a^n und
  - Ausgabe: n (in Binärdarstellung)
- **Lösung:**
  - Gehe von links nach rechts und
    - merke dabei, ob die Anzahl der a*s* bisher gerade oder ungerade war
    - Lösche dabei jedes ungerade a, d.h. das 1.,3.,5.,...
  - Schreibe 0 falls das Ergebnis gerade war und 1 sonst rechts an das Band
  - Falls mehr als ein a übrig bleibt
    - gehe ganz nach links und fange von vorne an.
- **Laufzeit:**
  - Jeder Durchlauf maximal 2n+2log n
  - Anzahl Durchläufe: log n
  - Ergibt: (2n+2log n)log n

    = 2nlog n+ 2(log n)^2= O(n log n)



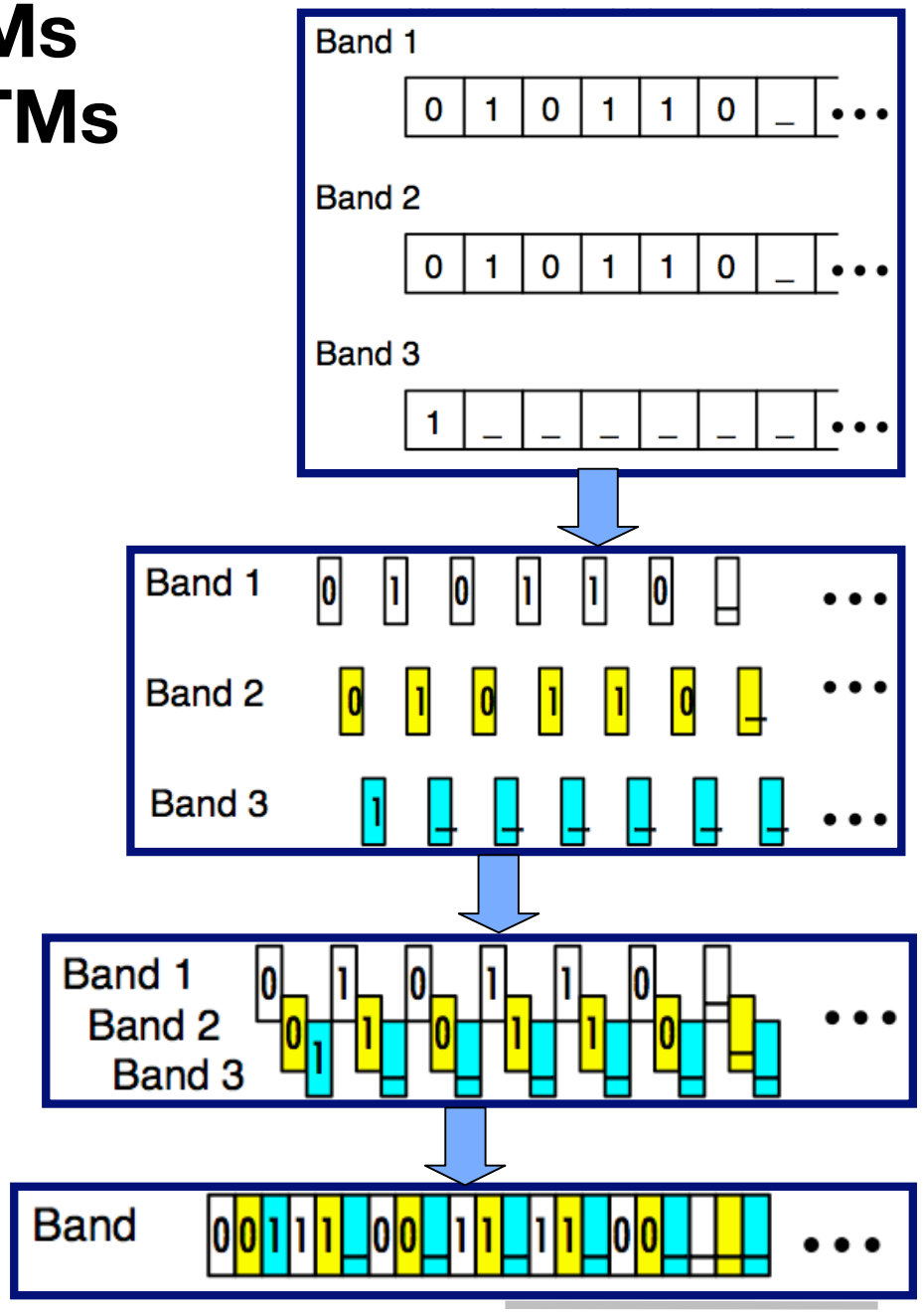
Effiziente Berechnung von $\{0^n1^n \mid n \geq 0\}$ mit 1-Band-TM

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

- **Algorithmus:**
  1. Zähle 0er
     - Sei x das Ergebnis
  2. Zähle 1er
     - Sei y das Ergebnis
  3. Vergleiche x und y
- **Laufzeit:**
  - 1. Schritt: O(n log n)
  - 2. Schritt: O(n log n)
  - 3. Schritt:
    - x und y sind binärkodiert und bestehen aus höchstens log n Zeichen
    - Der Vergleich zweier Zeichenketten der Länge m kann auf einer 1-Band-DTM in Zeit $O(m^2)$ durchgeführt werden
    - Ergibt $O((\log n)^2)$
- **Zusammengefasst: $O(n \log n) + O(n \log n) + O((\log n)^2) = O(n \log n)$**

k-Band-DTMs → 1-Band DTMs

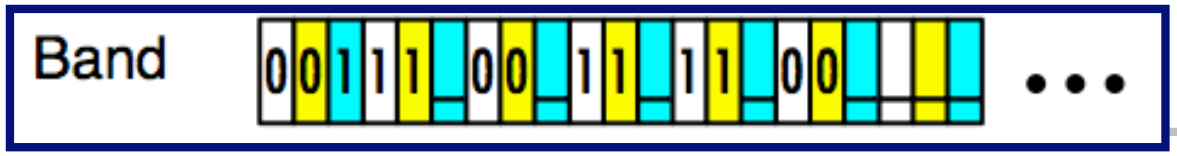
- **Theorem**
  - **$TIME_k(t(n)) \subseteq TIME_1(O(t^2(n)))$,** d.h.
  - Für $t(n)=\Omega(n)$ kann jede Berechnung einer k-Band-DTM von einer 1-Band DTM in Zeit $O(t^2(n))$ berechnet werden.
- **Beweis:**
  - Betrachte k-Band-DTM M mit Arbeitsalphabet Γ
  - und konstruiere 1-Band-DTM mit Alphabet $\Gamma \times \{\_, kopf\}$,
    - Speichere das i.-te Symbol von Band j an der Position j + i k.
    - Verwende Markierung *kopf* nur wenn der Kopf der k-Band-TM an der entsprechenden Stelle steht.
  - ...



k-Band-DTMs → 1-Band DTMs

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

- **Theorem**
  - **$TIME_k(t(n)) \subseteq TIME_1(O(t^2(n))$**
- **Beweis (Fortsetzung)**
  - Arbeitsweise 1-Band-DTM
    1. Kodiere Eingabe passend um
    2. Für jeden Schritt der k-Band-DTM
    3. Für jedes Band j
    4. Suche Kopfposition an den Stellen {j,j+k,j+2k,...}
    5. Lies Eingabe
    6. Berechne den Folgezustand, neues Symbol und Bewegungsrichtung
    7. Für jedes Band
    8. Suche Kopfposition
    9. Schreibe neues Zeichen
    10. Bewege Kopfmarkierung um k Zeichen nach links oder rechts
    11. Falls M hält, halte und akzeptiere/verwerfe entsprechend



k-Band-DTMs → 1-Band DTMs

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

- **Theorem**
  - **$\mathbf{TIME_k(t(n)) \subseteq TIME_1(O(t^2(n))}$**
- **Beweis (Laufzeit):**
  - Da die k-Band-DTM höchstens t(n) Schritte läuft, gibt es höchstens einen Folge k t(n) Zellen auf dem Band der 1-Band-DTM mit Symbolen ungleich "_"
  - Damit kann ein Kopf in Zeit k t(n) gefunden werden
  - Alle Köpfe werden in Zeit k^2 t(n) gefunden
  - Der eigentliche Simulationsschritt geschieht in konstanter Zeit.
  - Damit muss t(n) mal der Aufwand berechnet werden
    - für Köpfe finden und bewegen: $2\ k^2\ t(n) + k$
    - Simulationsschritt: 1
  - Das ergibt $(2\ k^2\ t(n) + k + 1)\ t(n) =$ **$O(t^2(n))$**,
    - da k eine Konstante ist

Akzeptanz einer NTM

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

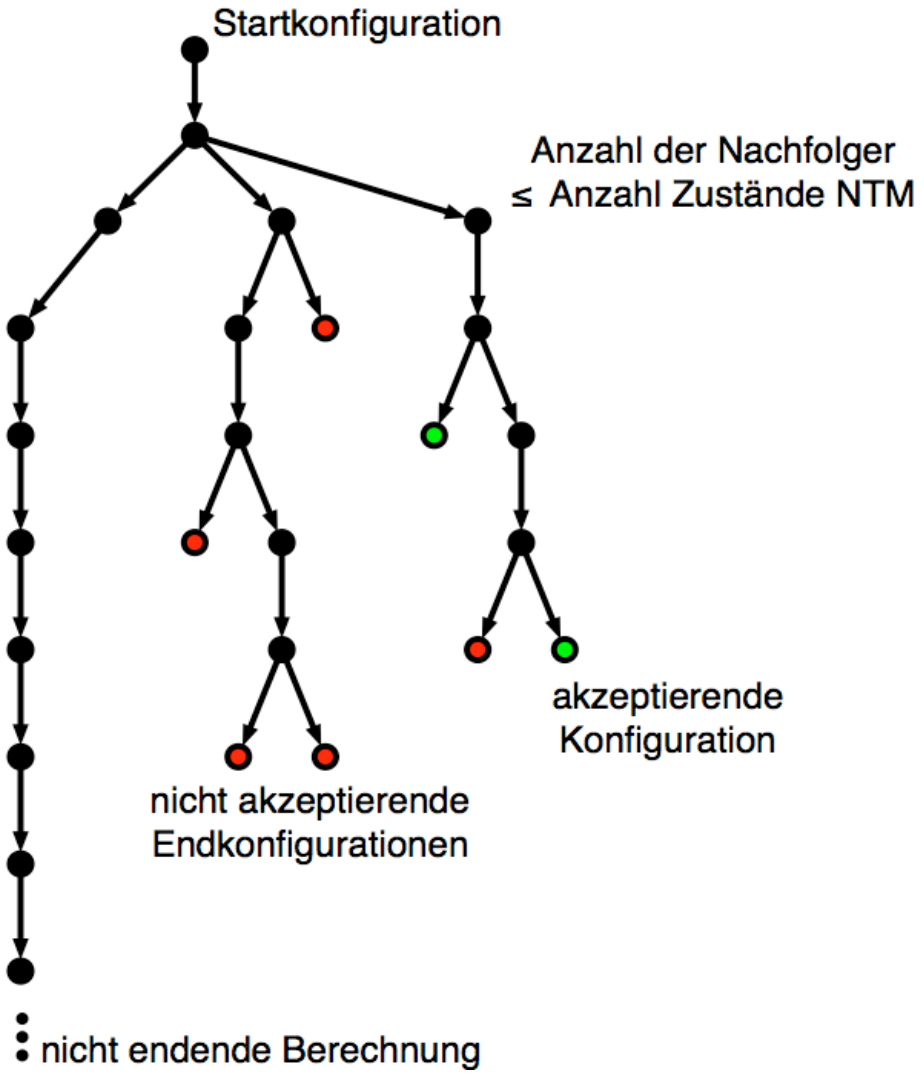
Eine Nichtdeterministische Turingmaschine M **akzeptiert** eine Eingabe w, falls es eine Folge von Konfigurationen C_1, C_2, …, C_k gibt, so dass

1. C_1 ist die Startkonfiguration von M bei Eingabe w
2. C_i kann zu C_{i+1} überführt werden
3. C_k ist eine akzeptierende Konfiguration

- Die von M akzeptierten Worte bilden die **von M akzeptierte Sprache** L(M)
- Eine NTM **entscheidet** eine Sprache, wenn jede Eingabe zu einem endlichen Berechnungsbaum der Konfigurationen führt.
- Die **Laufzeit** t(n) einer Entscheider-NTM ist die Länge des längstens Pfads im Berechnungsbaums für alle Eingaben der Länge n.

Berechnungsbaum einer NTM

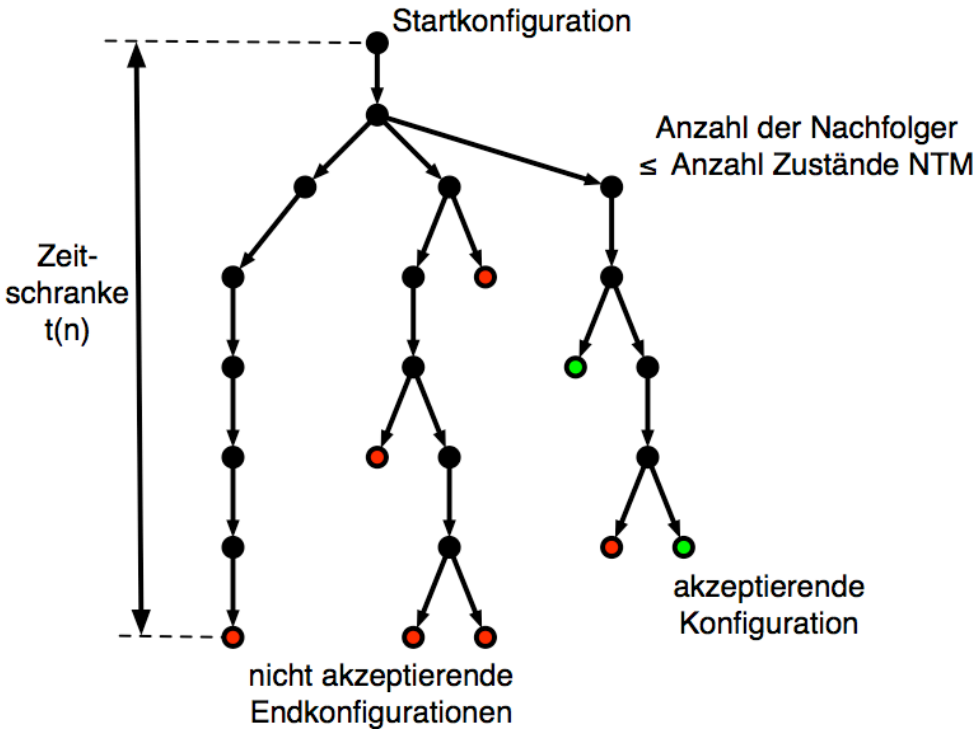
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



Nichtdeterministische Zeitkomplexitätsklassen

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

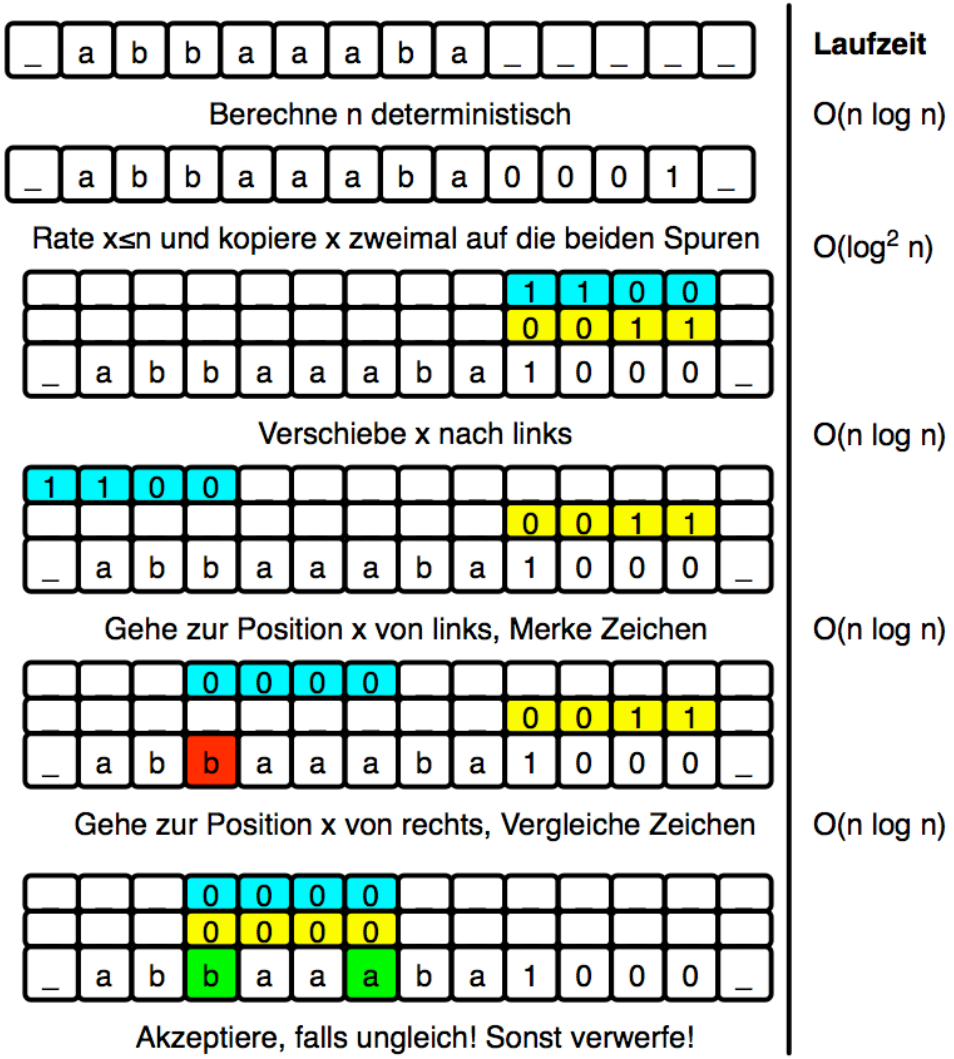
- **Definition**
  - Eine NTM ist t-Zeit-beschränkt, wenn für eine Eingabe der Länge n jede nichtdeterministische Berechnung höchstens t(n) Schritte benötigt.
- **Definition**
  - Sei $t:N \rightarrow R^+$ eine Funktion.
  - Die Zeitkomplexitätsklasse **NTIME(t(n))** ist die Vereinigung aller Sprachen,
    - die von einer **nicht**deterministischen O(t(n))-Zeit-Turing-Maschine entschieden werden.
  - Wird die Anzahl der Bänder auf k beschränkt, schreiben wir **$NTIME_{k\text{-}Band}(t(n))$** oder einfach **$NTIME_k(t(n))$.**



Beispiel

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

- ***Nicht-Palindrom*** **= {w ∈ {a,b}* | w≠w^{rev} }**
- **Es gilt:**
  Nicht-Palindrom **∈ NTIME$_{1\text{-Band}}$(n log n),**
  - da es eine O(n log n) -Zeit 1-Band-NTM die *Nicht-Palindrom* in Laufzeit O(n log n) löst.
- **Lösung:**
  - Rate Position x≤n
    - Dafür berechne zuerst deterministisch erstmal die Eingabegröße
  - Lies Zeichen an Position x und n-x
    - Speichere Positionszähler auf Extra-Spur
    - Verschiebe Positionszähler immer um eins
  - Merke das Zeichen jeweils im endlichen Speicher
  - Falls gilt $w_x \neq w_{n-x}$ akzeptiere, ansonsten verwerfe
- **Beachte: Es gilt**
  Nicht-Palindrom **∉ TIME$_{1\text{-Band}}$(n log n).**



Berechnung einer 1-Band-NTM durch eine k-Band-TM

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

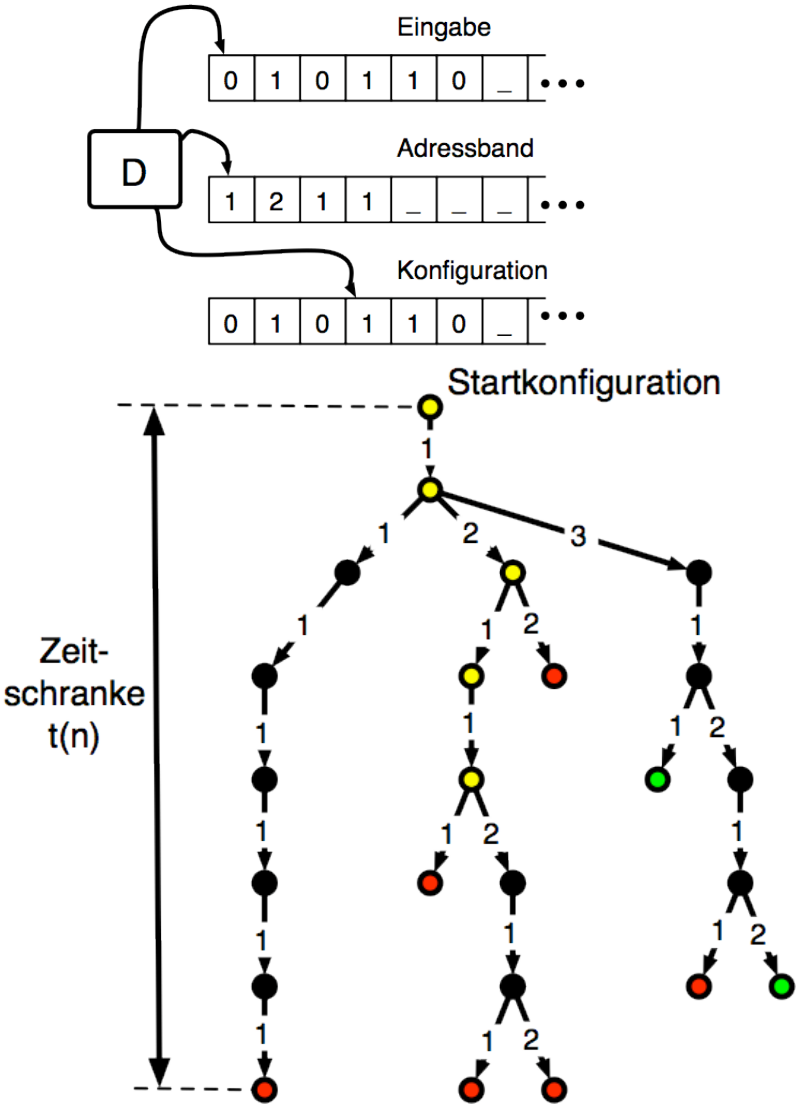
- **Theorem: Für $t(n)=\Omega(n)$**
  - **$NTIME_1(t(n)) \subseteq TIME_1(2^{O(t(n))})$, d.h.**
  - Jede Berechnung einer t-Zeit 1-Band-NTM kann von einer 3-Band DTM in Zeit $2^{O(t(n))}$ durchgeführt werden.
- **Beweis:**
  - Eine Konfiguration einer TM besteht aus
    - der Bandinschrift,
    - der markierten Kopfposition und
    - dem aktuellen Zustand
  - Simuliere 1-Band-NTM mit 3-Band-DTM
    - Speichere auf Band 1 die Eingabe
    - Speichere auf Band 2 die Position der NTM im Berechnungsbaum
    - Verwende Band 3 zum Speichern der aktuellen Konfiguration
  - **Lemma**:
    - Die Anzahl der Knoten im Berechnungsbaum einer t-Zeit NTM mit Zustandsmenge Q ist beschränkt durch $|Q|^{t(n)+1}$.
  - **Fakt**:
    - Jeder Konfigurationsübergang im Berechnungsbaum der NTM kann in einem Schritt auf dem dritten Band berechnet werden.
  - Die 3-Band-DTM wird dann mit einer 1-Band DTM berechnet.

Zeitbeschränkte Simulation einer NTM durch eine DTM

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

Beweis:

- Simulator DTM D:
  - Eingabeband
  - Adressband
  - Konfigurationsband
- Zähle per Tiefensuche alle Knoten des Berechnungsbaums im Adressband auf. Sortierung:
  - Lexikographisch
    - 1111111111
    - 1111111112
    - ...
    - 3333333333
- Für jeden Knoten des Berechnungsbaums:
  - Initialisiere Startkonfiguration
  - Führe dabei den (nun deterministischen) Konfigurationsübergang für jeden Pfad in Zeit t(n) durch
  - Falls simulierte NTM in akz. Zustand, dann halte und akzeptiere
- Falls jeder Knoten abgearbeitet, verwerfe



Ende der 17. Vorlesung

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

Christian Schindelhauer

Wintersemester 2006/07

17. Vorlesung

21.12.2006