Komplexitätstheorie

WiSe 2009/10 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

Komplexitätstheorie Gesamtübersicht

- Organisatorisches / Einführung
  Motivation / Erinnerung / Fragestellungen

- Diskussion verschiedener Komplexitätsklassen:
  Zeitkomplexität
  Platzkomplexität

- zugehörige Reduktionsbegriffe

- vollständige Probleme

- Anpassung von Klassenbegriffen und Reduktionen

Organisatorisches

Vorlesung: Montags 10-12 Uhr, H 6; Vorschlag ab 2. SW: 10.05-11.35
Zusätzlich in der ersten SW:
Freitag 10-12 Uhr, HZ 201

Übungen (Henning Fernau): Freitags 10-12 Uhr, HZ 201;
Beginn 2. Semesterwoche

Meine Sprechstunde: DO, 13-14 Uhr

Kontakt: fernau@uni-trier.de

Hausaufgaben / Schein ?! n.V. (Master ?! $\rightarrow$ mündliche Prüfung)

Erinnerung

In der letzten Übungsstunde wurden Mehrband-Turingmaschinen *in einer möglichen Formalisierung* behandelt.

Erkenntnis: Wir können mit der Hilfe von “Makros” fast auf “Hochsprachenniveau” Turingmaschinen programmieren.

$\leadsto$ Wir können auf etliche “Details” im Folgenden meist verzichten.

Wichtigste Rohstoffe Rechenzeit und Speicherplatz

Alternativen: Zahl von Prozessoren , Schaltkreisgröße/tiefe:
nur am Rande behandelt...Sei M (n. det.) k-Band-Turingmaschine

- Die *Zeitkomplexität* $T_M : W(\Sigma) \dashrightarrow \mathbb{N}$ von M ist definiert durch

$$T_M(x) := \min\{n \in \mathbb{N} \mid (\exists K \in \mathcal{K}_e) I_M(x) \xrightarrow[M]{n} K\}$$

- Die *Platzkomplexität* $S_M : W(\Sigma) \dashrightarrow \mathbb{N}$ von M ist definiert durch

$$S_M(x) := \min\{S(K) \mid K \in \mathcal{K}_e \wedge I_M(x) \xrightarrow[M]{*} K\}$$

Dabei ist $S(K) := \sum_{i=2}^{k-1} \lg(w_{i,l}) + 1 + \lg(w_{i,r})$ für Konfigurationen

$$K = (z, w_0, w_{1,l}, a_1, w_{1,r}, w_{2,l}, a_2, w_{2,r}, \ldots, w_{k-1,l}, a_{k-1}, w_{k-1,r})$$

Dabei

w_0	Inhalt des Ausgabebandes (Band 0)
$w_{1,l}a_1w_{1,r}$	Inhalt des Eingabebandes (Band 1)
$w_{i,l}a_iw_{i,r}$	Inhalt des Arbeitsbandes i
z	Zustand der Konfiguration K

Anmerkungen

—$T_M(x)$: kleinste Schrittzahl für Erreichen einer Endkonfiguration

—$S_M(x)$ minimaler Platzbedarf

Laut Formalisierung gilt einmal beschriebener Platz für immer als beschrieben.

—Platzbedarf ohne Eingabe-/Ausgabeband, nur ‘Hilfsspeicher’

—$T_M(x)$ und $S_M(x)$ genau dann definiert, wenn $x \in L_M$ gilt.

—det. TM: Mengen, deren Min. bestimmt wird, max. einelementig!

Beispiel 1 M mit $f_M(x) = xx^r$:

$$T_M(x) \leq 1 + 6 \cdot \lg(x) + 4 \cdot \lg(x) + 2$$

(pro Zeichen von x:
6 Schritte beim Kopieren von x, 4 Schritte beim Erzeugen von x^r)

$$S_M(x) \leq \lg(x) + 2$$

(Kopie von x und zwei Blanks).

Alternative Vorgehensweise:
—Keine Kopie der Eingabe auf Arbeitsband
—stattdessen Rückwärtslauf auf
$\Rightarrow$ Platzverbrauch lässt sich auf Null reduzieren.

Beispiel 2

M mit $L_M = \{xx^r \mid x \in W(\{0,1\})\}$

für jedes $x \in L$ stets

$$T_M(x) \leq 5 \cdot \frac{lg(x)}{2} + 6 \cdot \frac{lg(x)}{2} + 4$$

(pro Zeichen der Hälften von x:
je 5 Schritte beim Kopieren der ersten Hälfte von x
je 6 Schritte beim Vergleichen mit der zweiten Hälfte)

$$S_M(x) \leq \frac{lg(x)}{2} + 2$$

(Kopie der Hälfte von x und zwei Blanks).

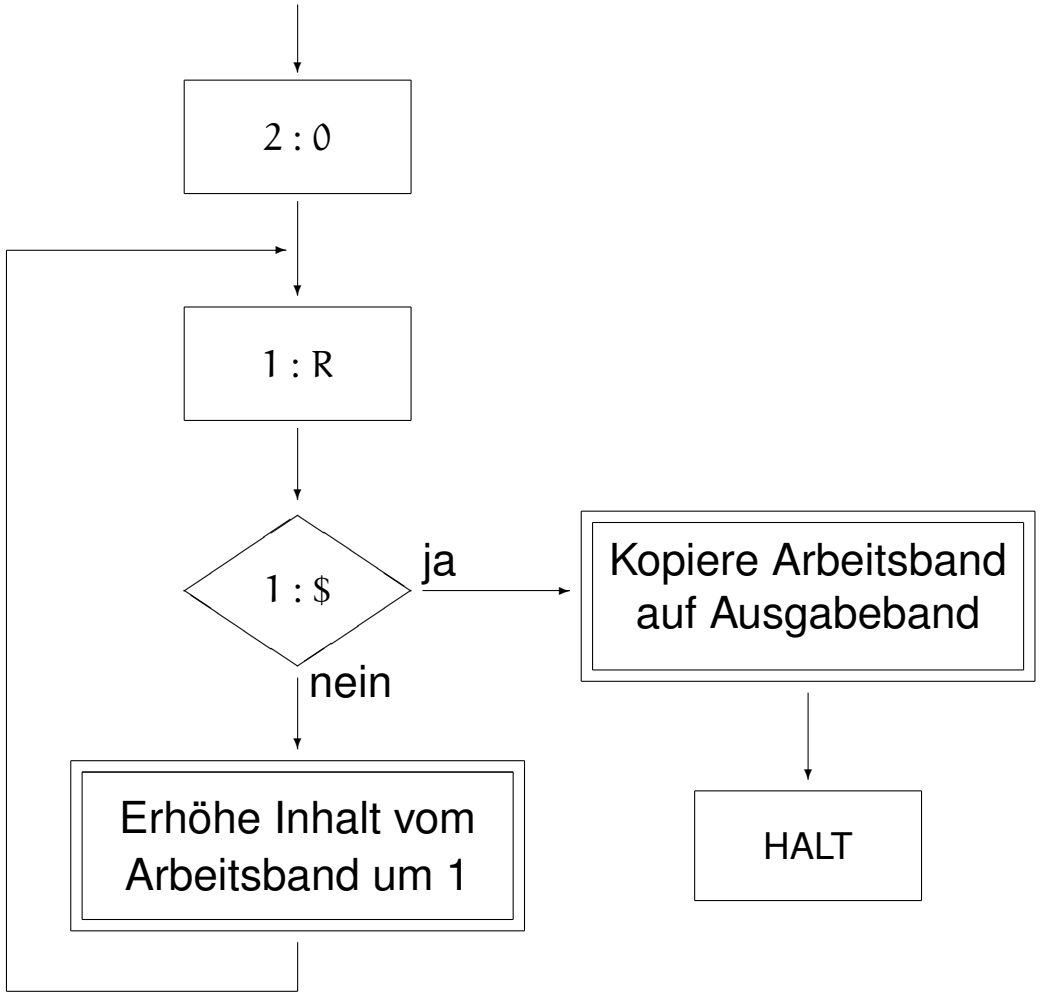
Beispiel 3 Binäres Zählen

f_{bin}: $W(\Sigma) \to W(\Delta)$ mit $\Sigma = \{1\}$, $\Delta = \{0, 1\}$ und

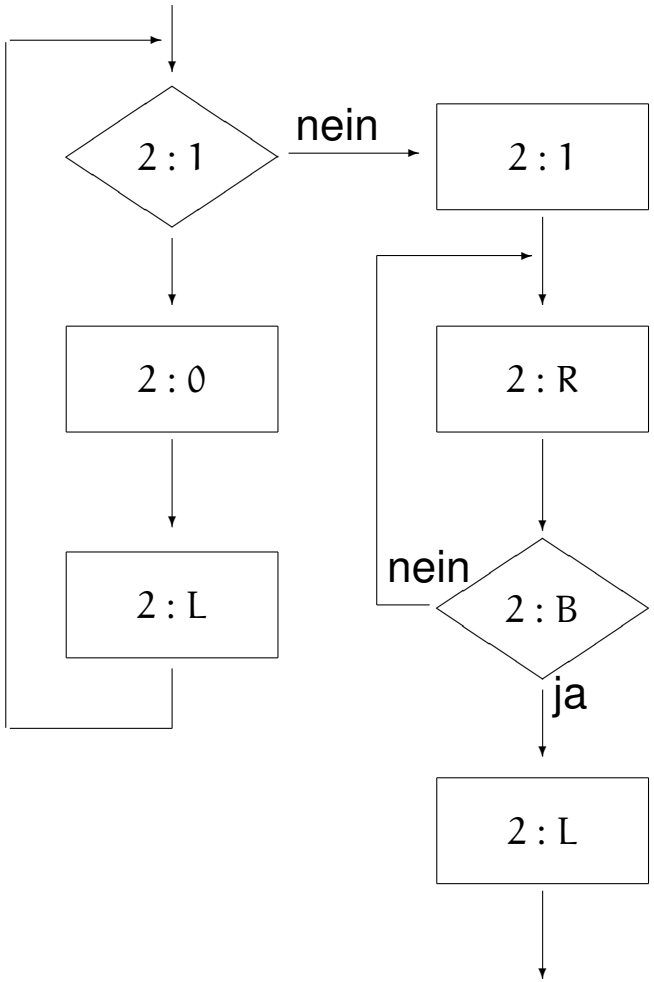
$$f_{bin}(x) = \text{ Binärdarstellung der Länge von } x$$

Im Folgenden: TM mit einem Arbeitsband und $\Delta = \{0, 1, B\}$

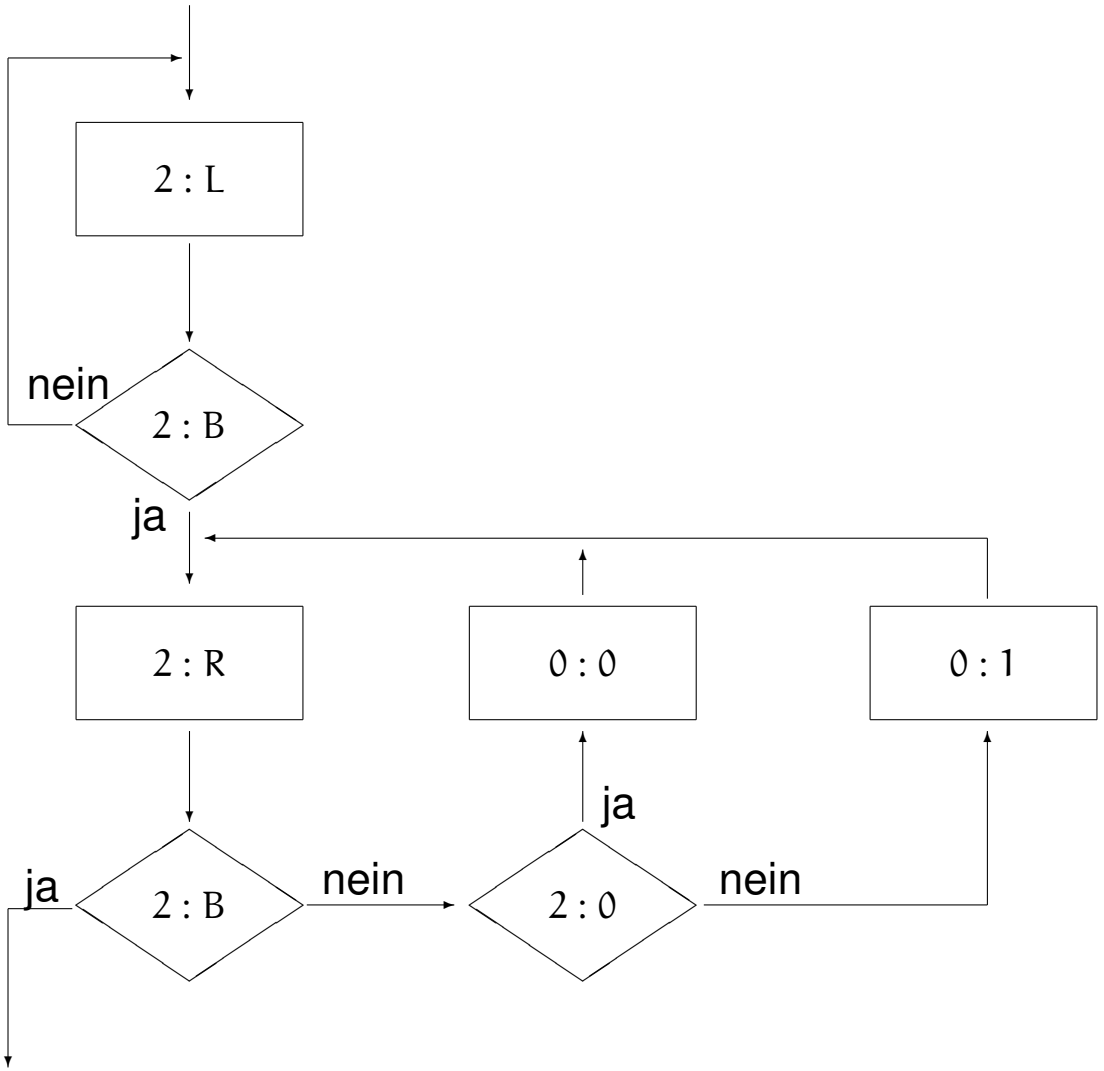
Erinnerung: Laut Formalisierung beschreibt $x : y$ im Flussdiagramm die Aktion y für Band x; speziell bedeutet $2 : 0$ in einer Raute den Test, ob der Kopf auf Band 2 gerade eine 0 liest, und $2 : 0$ in einem Rechteck formalisiert ein Schreiben der 0 auf Band 2.


2 : 0
1 : R
1 : $
ja
Kopiere Arbeitsband
auf Ausgabeband
nein
Erhöhe Inhalt vom
Arbeitsband um 1
HALT

Inkrementieren



Kopieren



Zur Komplexität

Im Spezialfall $w = \lambda$ $T_M(\lambda) = 11$.
Für $w \neq \lambda$ setze $n := lg(w)$, wähle $k \in \mathbb{N}_0$ minimal mit $n < 2^k \leq 2n$.
Dann gilt sicherlich: $T_M(w) \leq T_M(\bar{w})$ mit $lg(\bar{w}) = 2^k$.
Beim Inkrementieren von 0 bis 2^k treten dann folgende Fälle auf
(in Klammern die Zahl des Auftretens dieser Fälle):

$(\frac{1}{2} \cdot 2^k$-fach) letztes Zeichen wird 1

$(\frac{1}{4} \cdot 2^k$-fach) letztes Zeichen wird 0, vorletztes wird 1

$(\frac{1}{8} \cdot 2^k$-fach) zwei letzte Zeichen werden 0, drittletztes wird 1

...

Alternativ: Doppeltes Abzählen ...

$\Rightarrow$ Schranke für Gesamtaufwand zum Inkrementieren:

$$\sum_{i=1}^{\infty} \frac{1}{2^i} \cdot 2^k \cdot (5i) = 5 \cdot 2^k \sum_{i=1}^{\infty} \frac{i}{2^i} \overset{(*)}{=} 5 \cdot 2^k \cdot 2 = 10 \cdot 2^k \leq 20n$$

Zu $(*)$: Setze $f(p) := \sum_{i=0}^{\infty} p^i = \frac{1}{1-p}$ für $0 < p < 1$,

$\rightsquigarrow f'(p) = \frac{1}{(1-p)^2} = \sum_{i=1}^{\infty} i \cdot p^{i-1}$ (Ableitungstrick)

insbesondere also: $\sum_{i=1}^{\infty} \frac{i}{2^i} = \frac{1}{2} \sum_{i=1}^{\infty} i \left(\frac{1}{2}\right)^{i-1} = \frac{1}{2} \frac{1}{\left(1-\frac{1}{2}\right)^2} = 2$

Schranke für den Aufwand zur Kopieren: $6k + 2$ Schritte,
da n in binärer Notation aus k Zeichen besteht.

Insgesamt: $T_M(w) \leq 20 \lg(w) + 6 \log \lg(w) + 2$, d.h. lineare Zeit.

Elementare Zusammenhänge zwischen Zeit- und Platzbedarf:

Gegeben Konfigurationen-Folge $K_0, K_1, K_2, \ldots$ mit $K_{i-1} \xrightarrow{M} K_i$

Dann gilt stets $S(K_i) \leq S(K_{i-1}) + 1$, d.h. $S(K_i) \leq S(K_0) + i$.

Analog: Ausgabe wächst pro Schritt maximal um ein Zeichen.

Da bei k-Band-TM stets $S(I_M(x)) = k-2$ gilt, folgt:

Lemma 1 *Sei M eine (n.det.) k-Band-TM. Dann gilt für alle $x \in L_M$:*

$$S_M(x) \leq T_M(x) + k-2.$$

Ist M sogar deterministisch und damit die Funktion $f_M(x)$ definiert, so gilt zudem

$$lg(f_M(x)) \leq T_M(x).$$

Umkehrung: Platzverbrauch $\Rightarrow$ maximaler Zeitverbrauch:

Lemma 2 *Sei M eine (n.det.) k-Band-Turingmaschine.*

Dann gibt es Konstanten c_1, c_2, sodass für alle $x \in L_M$ gilt:

$$T_M(x) \leq c_1^{S_M(x)} \cdot lg(x) + c_2$$

Zwei Konfigurationen K, K' heißen *kongruent*, i.Z. $K \sim K'$, wenn sie sich höchstens im Inhalt des Ausgabebandes unterscheiden.
$\leadsto$ Kongruente Konfigurationen benutzen denselben Platz.
Beob.: Gilt $K \sim K'$ und $K \xrightarrow{M} K''$, so gibt es K''' mit $K'' \sim K'''$ und $K' \xrightarrow{M} K'''$.
Für det. TM gibt es in terminierender Konfig.folge keine zwei kongruenten Konfig., da daraus mit Beob. folgt: Die Endkonfig. der Folge hätte einen Nachfolger.
#Kongruenzklassen bei Eingabe $x \leq |\Gamma|^{S_M(x)} * \#\text{Kopfpositionen} * |Z|$
Im nichtdet. Fall Beschränkung auf "kürzeste Konfig.folgen" möglich, bei deren "Konstruktion" Schleifen zwischen kongruenten Konfigurationen weggelassen werden können.

Manipulationen am Arbeitsalphabet:

In Definition der TM: Arbeitsalphabet Γ beliebig groß
Zunächst z.z.: Beschränkung auf zwei Zeichen unwesentlich!

Lemma 3 (Reduktion des Arbeitsalphabetes) *Zu jeder k-Band-TM M gibt es eine k-Band-TM M' und Konstanten c_1, c_2 mit*

$$L_M = L_{M'}, \quad S_{M'}(x) \leq c_1 \cdot S_M(x), \quad T_{M'}(x) \leq c_2 \cdot T_M(x)$$

für alle $x \in L_M$, wobei das Arbeitsalphabet Γ' von M' nur zwei Elemente hat.
Ist M deterministisch, so gibt es auch ein deterministisches M' mit diesen Eigenschaften, für das zusätzlich für die berechneten Funktionen $f_M = f_{M'}$ gilt.

Beweisidee: Führe binäre Blockcodes ein und erweitere Zustandsalphabet.

Wie viele Bänder brauchen wir ?

In Definition der TM: beliebig viele Arbeitsbänder zugelassen

Lemma 4 (Reduktion der Zahl der Arbeitsbänder)

Zu jeder det. k-Band-TM M gibt es eine det. 3-Band-TM M' und eine Konstante c mit

$$f_M = f_{M'}, \quad S_{M'}(x) \leq S_M(x), \quad T_{M'}(x) \leq c \cdot T_M(x) \cdot S_M(x)$$

für alle $x \in L_M$.

Zwei Konstruktionen möglich:
(1) Schreibe die Arbeitsbänder (mit Trenner) hintereinander auf ein Band.
(2) Richte auf dem einen Arbeitsband mehrere *Spuren* ein: Die Kopfposition der simulierenden Maschine ist insofern fix, als dass sie nach einer Einzelschrittsimulation stets alle Bandinhalte unter allen Köpfen der simulierten Maschine “gleichzeitig sieht”. Kopfbewegungen der simulierten Maschine werden daher durch “Spurbewegungen” der simulierenden Maschine nachgeahmt.

Wie viele Bänder brauchen wir ?

Modifiziertes Resultat bei n.det. TM:
minimale Zeit $T_M(x)$ und minimalen Platz $S_M(x)$ können aus verschiedenen Rechenverläufen stammen!

Folgerung 5 *Zu jeder n.det. k-Band-TM M gibt es eine n.det. 3-Band-TM M' und eine Konstante c mit*

$$L_M = L_{M'}, \quad S_{M'}(x) \leq S_M(x), \quad T_{M'}(x) \leq c \cdot T_M(x)^2$$

für alle $x \in L_M$.

Anmerkungen

(1) Simulation einer beliebigen festen Zahl von Arbeitsbändern mit zwei Arbeitsbändern:

$$c \cdot T_M(x) \cdot \log T_M(x) + c$$

(Hennie/Stearns, 1966).

(2) Später: konstanter Faktor bei Speicherbedarf oder quadrierter Zeitbedarf meist unwesentlich

(3) Nachweis oberer Komplexitätsschranken:
k-Band- TM mit beliebigem Γ und beliebigem (festem) k erlaubt

(4) Beim Nachweis unterer Schranken:
3-Band-TM mit zweielementigem Γ ausreichend

Abschluss des Kapitels über Turingmaschinen:

Techniken beim Umgang mit Turingmaschinen

$\Rightarrow$ Erleichterung der Konstruktion von Maschinen

$\Rightarrow$ Hilfe bei Abschätzung der Komplexität.

Speicherung von Zahlen:

$\mathbb{N}$ meist binär als $w \in W(\{0,1\})$ und ohne führende Nullen

$\mathbb{N}$ unär: w über beliebigem Alphabet stellt Zahl $\lg(w)$ dar.

—Speicherung auf einem Arbeitsband mit 'Endemarkern' #, \$ oder Blanks

—Speicherplatz: Unär: $n+2$ Zeichen, binär $\log n + 2$ Zeichen

dabei zur Vereinfachung:

$$\log n := \begin{cases} 1 & \text{falls } n \in \{0,1\}, \\ \min\{i \in \mathbb{N} \mid 2^i > n\} & \text{sonst} \end{cases}$$

—Zahl der Arbeitsbänder unwesentlich

$\Rightarrow$ o.B.d.A. jede Zahl binär auf eigenem Band

(Speicherung von Zahlen:)

$\mathbb{Z}$: natürliche Zahl mit Vorzeichen

$\mathbb{Q}$: Paare ganzer Zahlen (Zähler/Nenner)

$\mathbb{R}$: Problematisch...($\Rightarrow$ Vorlesung Rechnerarithmetik)

Hier: meist nur $\mathbb{N}$...

Arithmetische Operationen:

—Hochsprachen-Operation $Z := X + Y$ übersetzt auf 3 Bänder

—Alter Inhalt der Z-Bandes wird überschrieben

—Endemarker: 'alter' Bandinhalt stört nicht

—Speicherzellen werden wiederverwendet

—Zeichen gleicher Wertigkeit an gleiche Stelle des Bandes
$\Rightarrow$ TM braucht zur Speicherung von Zahlen in einer solchen 'Variablen' soviel Platz wie für die größte überhaupt darin gespeicherte Zahl.

—O.B.d.A. *vor und nach* jedem Zugriff auf Zahl $x_n \ldots x_0$ steht Lesekopf auf x_0

(Arithmetische Operationen)

—Elementare Arithmetik:

Addition, Subtraktion, Multiplikation und Division (mit Rest) ohne ‘Hilfsspeicher’ möglich

Operation	Zeit
$Z := X + Y$	$c \cdot \max\{\log X, \log Y\}$
$Z := X - Y$	$c \cdot \max\{\log X, \log Y\}$
$Z := X \cdot Y$	$c \cdot \log X \cdot \log Y$
$Q := X/Y, R = X \bmod Y$	$c \cdot \log X \cdot \log Y$

hier: ‘Schulalgorithmen’ zum Multiplizieren und Dividieren

Bessere Zeitschranken: Vorlesung Rechnerarithmetik

Felder:

—Felder (natürlicher Zahlen) mit *variabler* Dimension n auf *einem* Band möglich

—verzahnte Speicherung: n Worte $w_i = x_{i,1} \dots x_{i,k}$ gleicher Länge k in Reihenfolge

$$x_{1,1} \dots x_{n,1} x_{1,2} \dots x_{n,2} \dots\dots x_{1,k} \dots x_{n,k}$$

—gleiche Länge über Endemarker

—Kopf des Bandes am rechten Ende der ersten Zahl (auf $x_{1,k}$)

—Zugriff auf Feld mit zusätzlicher Zähl-Variable

—Bsp.: Zugriff $F(Y) := Z$ bei Dimension X des Feldes F dauert $c \cdot (Y + X \cdot \log Z)$

—Bsp.: Zugriff $Z := F(Y)$ dauert $c \cdot (Y + X \cdot \log F(Y))$

Beachte bei den letzten beiden Punkten: Endemarker für einzelne Zahlen

(Felder)

—Analog:
Speicherung von Worten unterschiedlicher Länge über Σ
mit Endemarkern ($\notin \Sigma$)
Komplexität des Zugriffes: $\lg(Z)$ statt $\log Z$

Markieren von Speicherzellen:

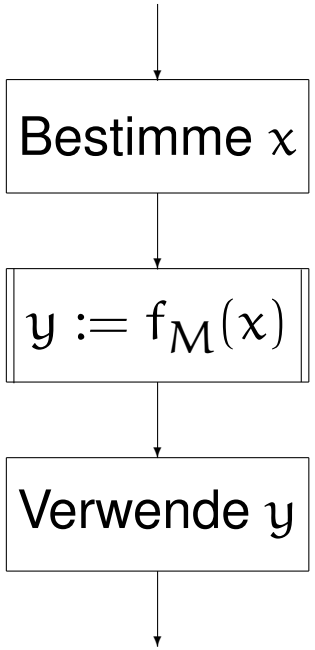
—Oft genutzter Trick: Arbeitsbänder in zwei Spuren unterteilen
—eine Spur zur Speicherung von Daten
—zweite Spur zur Markierung von Positionen auf dem Band
$\Rightarrow$ z.B. möglich: alle Zeichen eines Bandes löschen und trotzdem wieder Ausgangsposition einnehmen

Unterprogramme

M det. k-Band-TM

$\Rightarrow f_M$ bei anderer TM M' verwendbar

wie Funktion in höherer Programmiersprache:



(Unterprogramme)

—$\mathcal{M}'$ benutzt k Bänder $B_0, \ldots, B_{k-1}$ zur 'Simulation' von $\mathcal{M}$

—Band B_1 wird mit '#x$' initialisiert

—Band B_0 wie Ausgabeband (nach Schreiben Kopf nach rechts)

—Bänder B_2 bis B_{k-1} als Arbeitsbänder für die Simulation

'Simulation':

—alle Zustände von $\mathcal{M}$ in $\mathcal{M}'$ übernommen

—Befehle von $\mathcal{M}$ in Befehle für Bänder B_i umgesetzt

—Am Ende alle benutzten Bänder löschen
—Köpfe wieder an Ausgangsposition zurück
$\Rightarrow$ Bei wiederholter Simulation Speicherplatz wiederverwendet!

(Unterprogramme)

Zeitaufwand für Simulation $y := f_M(x)$:

$$c \cdot (lg(x) + t_M(x))$$

Platzbedarf:

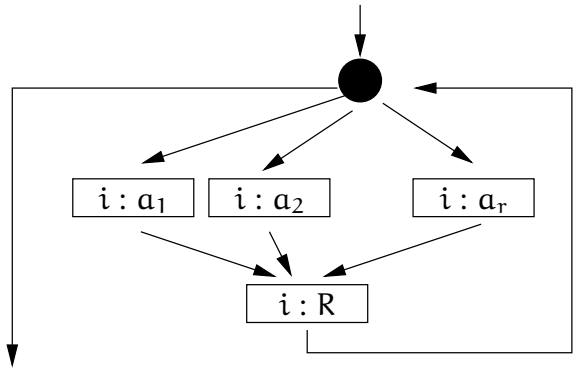
$$c \cdot (lg(x) + s_M(x) + lg(f_M(x))$$

(inklusive Kopieren von x auf Band B_1 und
Kopieren von $f_M(x)$ von Band B_0 auf anderes Band von M')

Anmerkung: Später andere Methode der Simulation, die weniger Speicher braucht!

Raten eines Wertes:

‘ **Wähle beliebiges Wort w über Alphabet $\{a_1, \ldots, a_r\}$**’



—Analog: ‘ **Wähle eine beliebige natürliche Zahl**’
—Mit zusätzlichen Zählern leicht:
‘ **Wähle beliebige natürliche Zahl zwischen n und m**’
oder
‘**Wähle beliebiges Wort w der Länge l**’

Speedup ?!

Erinnerung: Reduktion der Arbeitsalphabetes:

konstruierte TM 'langsamer', braucht mehr Platz als Original-TM

Umkehrung: 'Beschleunigung' von TMs durch Zusammenfassung von Schritten:

Lemma 6 (linearer Speedup / lineare Kompression) *Zu jeder TM* M *und jedem* $\varepsilon > 0$ *gibt es TM* M' *und* M'' *mit* $L_M = L_{M'} = L_{M''}$ *und*

$$t_{M'}(x) \leq c \cdot (|x|+1) + t_M(x) \cdot \varepsilon$$

und

$$s_{M''}(x) \leq s_M(x) \cdot \varepsilon + 2$$

Konstante c *nur von* Σ *abhängig!*

Speedup ?!

Grundidee: Zusammenfassen benachbarter Zeichen zu einem “Macrozeichen” (Aufblähen des Bandalphabetes).

Achtung: Zeitspeedup von exakter Definition des TM-Modells abhängig!
Notwendig: TM kann in einem Schritt Zeichen ‘einlesen’
Bei unserem Modell gilt dieser Teil des Lemma daher nicht!

$\Rightarrow$ Angabe von Faktoren bei Komplexitätsangabe unsinnig
(oder: Bandalphabet fixieren!)