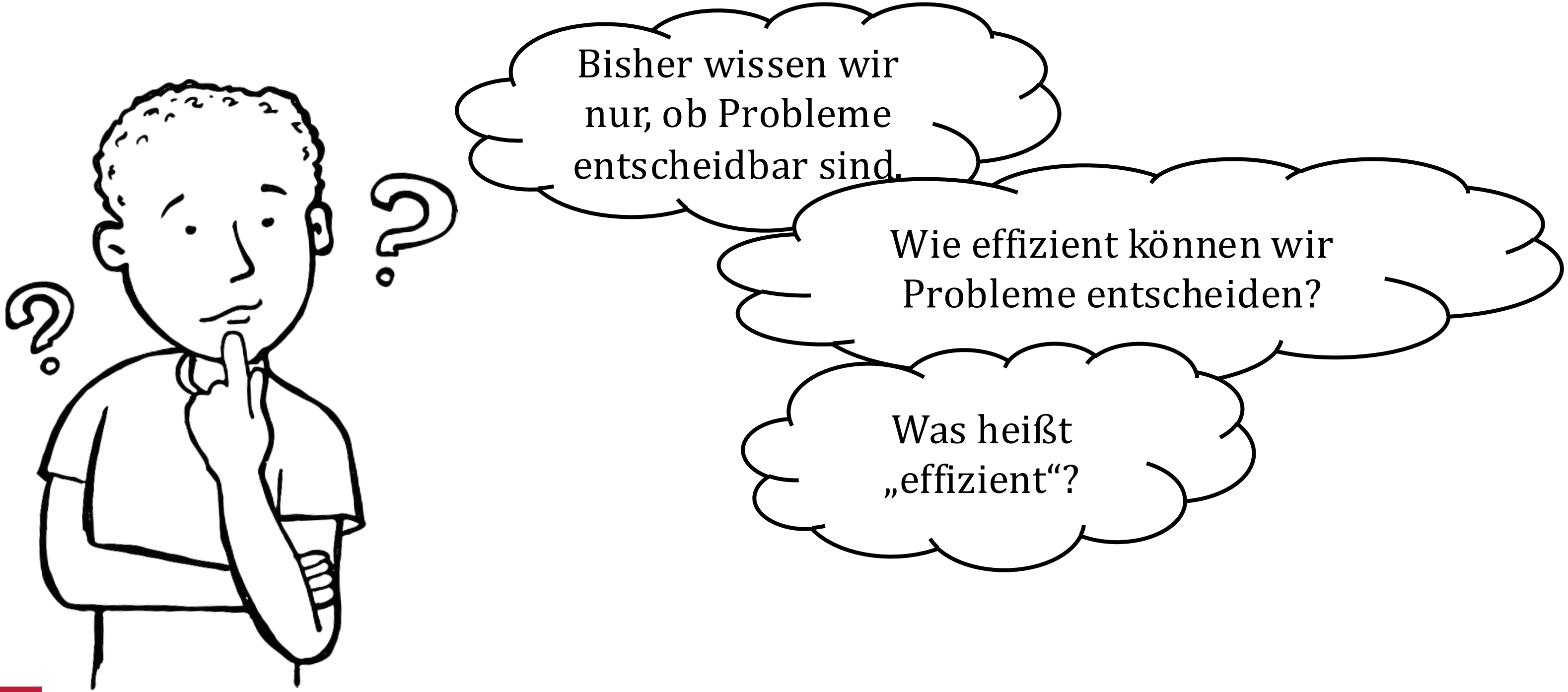
Technische Universität Braunschweig

Theoretische Informatik 2

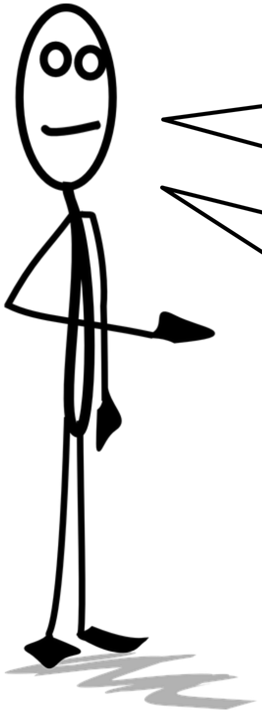
Arne Schmidt

Kapitel 7 – Grundlagen der Komplexitätstheorie

Effizienz


Bisher wissen wir nur, ob Probleme entscheidbar sind.
Wie effizient können wir Probleme entscheiden?
Was heißt „effizient“?

Laufzeit und Platzverbrauch



Wir betrachten Klassen von Problemen, die bzgl. der Laufzeit oder des Speicherbedarfs zum Lösen ähnlich sind.

Die Ähnlichkeit messen wir über die Landau-Notation!

Kapitel 7.1 – Landau-Notation

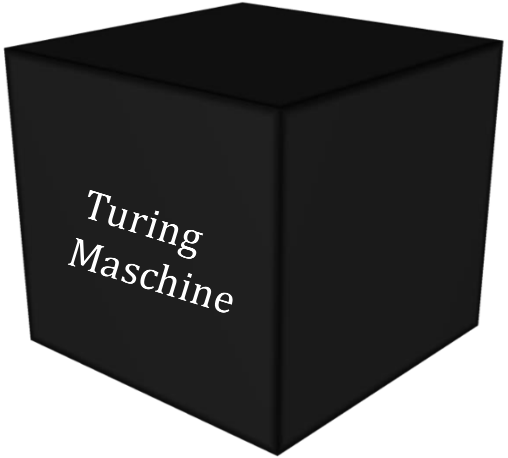
Vergleichen von Laufzeiten

Die Turing-Maschine benötigt für ein Wort der Länge n
$17n^3 + n^2 - 200n + 300$ Schritte.

Eine andere Turing Maschine benötigt für ein Wort der Länge n
$2n^3 - 3n^2 + n + 1$ Schritte.

Beides sind Funktionen $\mathbb{N} \to \mathbb{N}$, aber wachsen unterschiedlich.
„Nach hinten raus“ (für Große n) veralten die sich sehr ähnlich!

Hier interessant ist nur der Teil n^3.



O-Notation

Definition 7.1
Zu einer Funktion $f\colon \mathbb{N} \to \mathbb{N}$ ist $O(f)$ die Klasse der Funktionen, die **asymptotisch kleiner-gleich** f sind:

$$O(f) = \{\, g\colon \mathbb{N} \to \mathbb{N} \mid \exists n_0, m \in \mathbb{N}, \forall n \geq n_0\colon g(n) \leq m \cdot f(n) \,\}$$

Beispiele 7.2

a) $O(1) = O(5000) = O(k)$ für alle Konstanten $k \in \mathbb{N}$.

b) $O(\log n)$ ist die Klasse der logarithmischen Funktionen. Bspw. gilt $1 \in O(\log n)$, aber $\log n \notin O(1)$

c) $O(n), O(n^2), O(n^3)$, usw. sind die Klasse der linearen, quadratischen, kubischen, ... Funktionen. Zu jedem Polynom $p(n) = a_k n^k + a_{k-1} n^{k-1} + \cdots + a_1 n + a_0$ gilt $p(n) \in O\left(n^k\right)$.

d) $\left\{ n \mapsto 2^{f(n)} \mid f(n) \in O(n) \right\}$ ist die Klasse der exponentiellen Funktionen mit lin. Exponenten. Kurzschreibweise für solche Funktionen: $2^{O(n)}$

Bemerkung 7.3

In der Literatur wird statt „∈“ auch gelegentlich „=“ verwendet.

Achtung:
Dann ist zwar $n = O(n^2)$, aber weiterhin nicht $O(n) = O(n^2)$.

Kapitel 7.2 – Komplexitätsklassen

Zeitverbrauch

Definition 7.4
Sei M eine TM (potentiell nicht-det., potentiell mit mehreren Bändern). Sei $x \in \Sigma^*$ eine Eingabe für M. Der **Zeitverbrauch** (oder Rechenzeit) von M für Eingabe x ist

$$\mathrm{Time}_M(x) = \max\{\, \min\{\, i \mid c_i \text{ Haltekonfiguration} \,\} \mid c_0 \to c_1 \to \cdots \text{ Berechnung von } M \text{ zu } x \,\}$$

$\mathrm{Time}_M(x) = \infty$, wenn M eine nicht-haltende Berechnung zur Eingabe x hat.
Wir gehen davon aus, dass alle betrachteten TMs Entscheider sind und $\mathrm{Time}_M(x)$ liefert eine natürliche Zahl. D.h. Es ist eine Funktion $\Sigma^* \to \mathbb{N}$.

Zeitkomplexität

Definition 7.5
Zu einer TM M ist die **Zeitkomplexität** $\mathrm{Time}_M : \mathbb{N} \to \mathbb{N}(\cup \{\infty\})$ die Funktion mit

$$\mathrm{Time}_M(n) = \max\{\, \mathrm{Time}_M(x) \mid x \in \Sigma^*, |x| = n \,\}$$

„Worst-Case für eine TM bei Eingaben der Länge n.“

Definition 7.6
Sei $f : \mathbb{N} \to \mathbb{N}$ eine Funktion. Wir sagen, dass M $\boldsymbol{f}$**-zeitberschränkt** ist, wenn $\mathrm{Time}_M(n) \leq f(n)$ für alle $n \in \mathbb{N}$.

Bemerkung 7.7
Ist M f-zeitbeschränkt für irgendeine Funktion, dann muss M ein Entscheider sein.

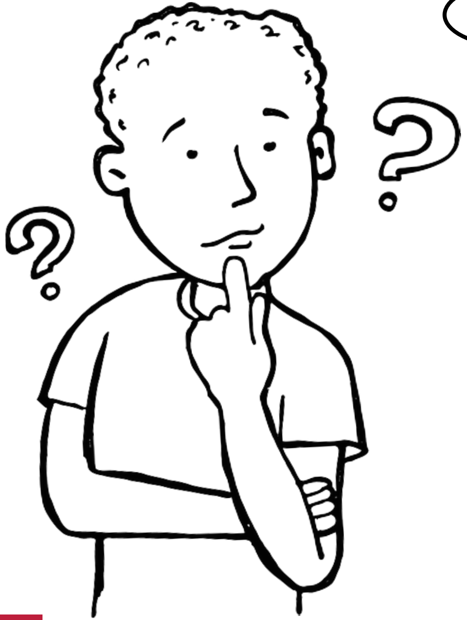
Zeitkomplexität

Definition 7.8 (Grundlegende Zeitkomplexitätsklassen)
Sei $f : \mathbb{N} \to \mathbb{N}$ eine Zeitschranke und $m \in \mathbb{N}, m > 0$. Wir definieren die grundlegenden Zeitkomplexitätsklassen

$$\mathrm{DTIME}_m(f) = \{\, \mathcal{L}(M) \mid \text{M ist eine } m\text{- Band DTM } und\ f\text{- zeitbeschränkt} \}$$
$$\mathrm{NTIME}_m(f) = \{\, \mathcal{L}(M) \mid \text{M ist eine } m\text{- Band NTM } und\ f\text{- zeitbeschränkt} \}$$

Bemerkung 7.9
Für 1-Band-Maschinen lassen wir den Index weg.

Platzverbrauch



Wie sieht das mit Platzverbrauch aus?

Weniger als die Eingabe-größe geht schlecht.

Können wir eher betrachten, wie viel **zusätzlich** benötigt wird?

Read-Only Eingabe

Definition 7.10
Eine **TM mit read-only Eingabe und *m*** $\in \mathbb{N}$ **Arbeitsbändern** ist eine (m+1)-Band-TM, bei der die Eingabe nicht verändert werden kann.
Formal ist jede Transitionsregel der Form

$$\delta(q, \underset{\text{Nicht-überschreibbares Symbol}}{a}, \underbrace{b_1, \dots, b_m}_{\text{Beliebig überschreibbare Symbole}}) \ni (q', a, c_1, \dots, c_m, \underbrace{d, d_1, \dots, d_m}_{\text{Bewegung auf den } m+1 \text{ Bändern}})$$

Platzverbrauch

Definition 7.11
Sei M eine TM mit read-only Eingabeband und m Arbeitsbändern (potentiell nicht-deterministisch). Sei $x \in \Sigma^*$ eine Eingabe für M.

a) Die Länge $|w|$ des Inhalts eines Arbeitsbandes in einer Konfiguration von M ist die Länge des belegten Bandinhalts. $\sqcup$-Symbole werden nicht mitgezählt.
b) Sei c eine Konfiguration von M. Der **Platzverbrauch von M in Konfiguration c** ist
$$\mathrm{Space}_M(c) = \max\{|w| \mid w \text{ ist der Inhalt eines Arbeitsbandes in Konfiguration } c\}$$
c) Der Platzverbrauch von M zu Eingabe x ist
$$\mathrm{Space}_M(x) = \max\{\mathrm{Space}_M(c) \mid c \text{ ist Konfiguration in einer Berechnung von } M \text{ zu Eingabe}\}$$

Ist der Platzverbrauch von M auf x unberschränkt, schreiben wir $\mathrm{Space}_M(x) = \infty$.

Platzkomplexität

Definition 7.12
Zu einer TM M ist die **Platzkomplexität** $\text{Space}_M : \mathbb{N} \to \mathbb{N}(\cup \{\infty\})$ die Funktion mit

$$\text{Space}_M(n) = \max\{\, \text{Space}_M(x) \mid x \in \Sigma^*, |x| = n \,\}$$

„Worst-Case für eine TM bei Eingaben der Länge n."

Bemerkung 7.13
Ist man nur an mind. Linearem Platzbedarf interessiert, kann man die read-only Bedingungen weglassen.

Definition 7.14
Sei $f : \mathbb{N} \to \mathbb{N}$ eine Funktion. Wir sagen, dass M **f-platzberschränkt** ist, wenn $\text{Space}_M(n) \leq f(n)$ für alle $n \in \mathbb{N}$.

Bemerkung 7.15

Ist M ein Entscheider, ist $\mathrm{Space}_M\colon \mathbb{N} \to \mathbb{N}$, liefert also nie ∞. (Pro Zeitschritt können wir nur ein Zeichen pro Band schreiben.)

Andersherum gilt das nicht unbedingt:
Eine TM kann endlich viel Platz belegen, muss aber nicht halten; ist dann also kein Entscheider.

Platzkomplexität

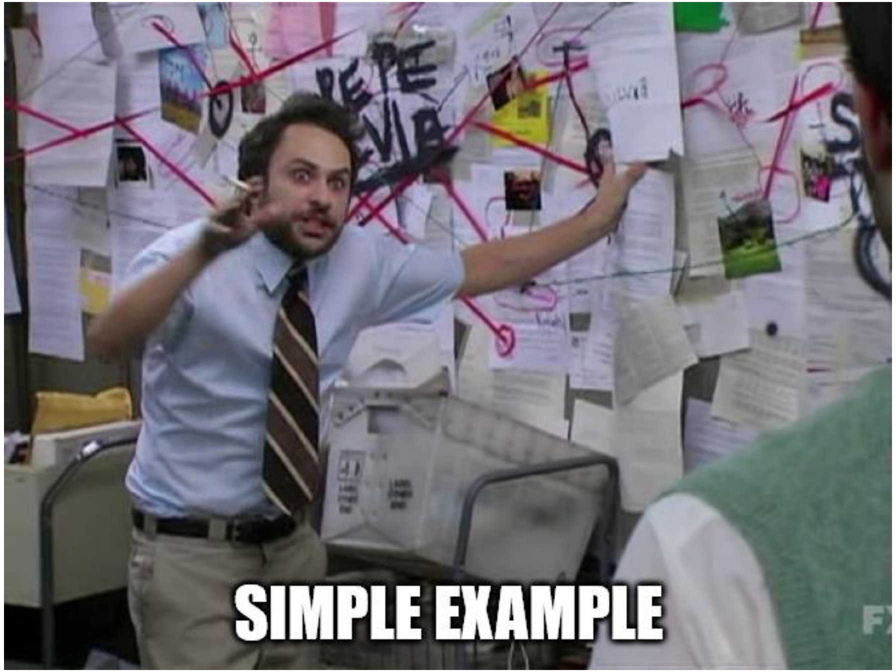
Definition 7.8 (Grundlegende Platzkomplexitätsklassen)
Sei $f\colon \mathbb{N} \to \mathbb{N}$ eine Zeitschranke und $m \in \mathbb{N}$. Wir definieren die grundlegenden Platz-komplexitätsklassen

$$\mathrm{DSPACE}_m(f) = \left\{ \mathcal{L}(M) \;\middle|\; \begin{array}{c} M \text{ ist DTM mit read-only Eingabe, } m \text{ Arbeitsbändern,} \\ \text{ein Entscheider und } f\text{-platzbeschränkt} \end{array} \right\}$$

$$\mathrm{NSPACE}_m(f) = \left\{ \mathcal{L}(M) \;\middle|\; \begin{array}{c} M \text{ ist NTM mit read-only Eingabe, } m \text{ Arbeitsbändern,} \\ \text{ein Entscheider und } f\text{-platzbeschränkt} \end{array} \right\}$$

Bemerkung 7.9
Für 1-Band-Maschinen lassen wir den Index weg.

Beispielzeit!



Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

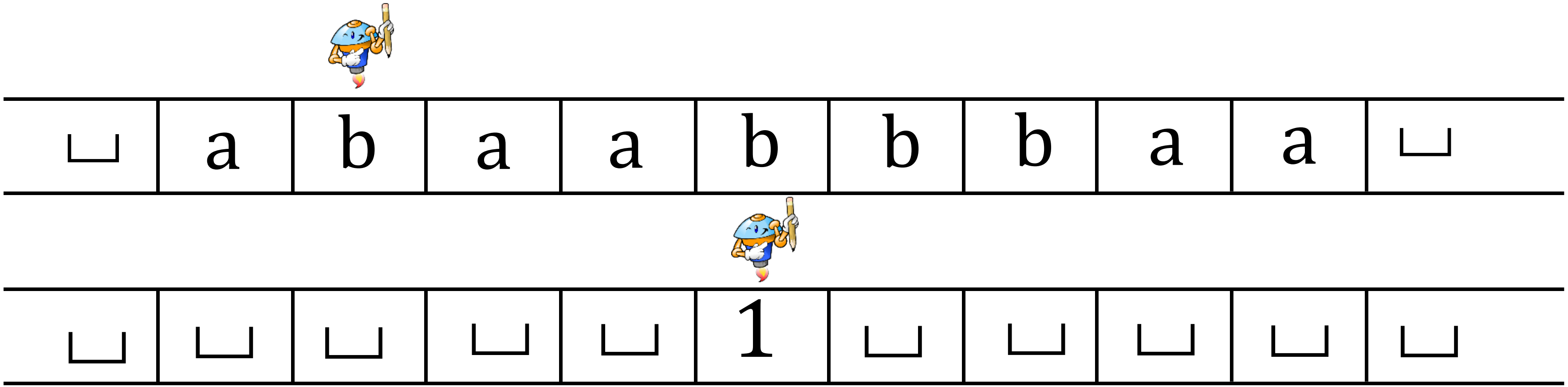
␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	␣	␣	␣	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.

Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$



Beispiel 7.17

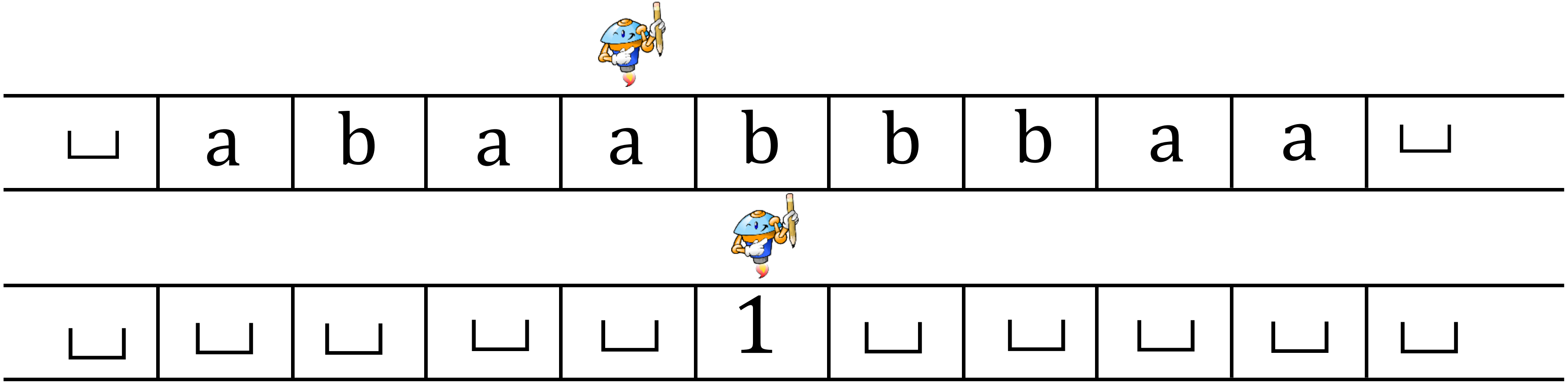
Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	␣	␣	0	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$



Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	␣	1	0	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	␣	0	1	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	␣	0	0	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	−	0	1	␣	␣	␣	␣	␣

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.

Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	+	0	0	␣	␣	␣	␣	␣

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

␣	a	b	a	a	b	b	b	a	a	␣

␣	␣	␣	+	0	1	␣	␣	␣	␣	␣

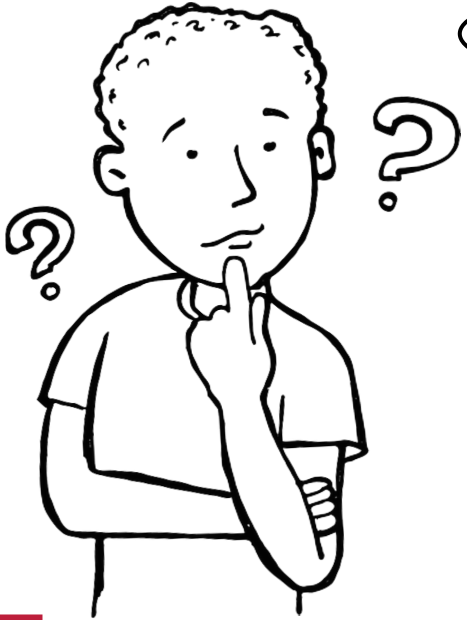
Wort nicht akzeptiert!

Beispiel 7.17

Sei $\mathcal{L} = \{\, x \in \{a, b\}^* \mid \text{Anzahl an } a \text{ und } b \text{ in } x \text{ ist gleich} \,\}$.
Zeige: $\mathcal{L} \in DSPACE\big(O(\log n)\big)$

Zähle also auf dem Arbeitsband +1 für jedes a, -1 für jedes b.
Zähler ist im Bereich $[-n, n]$, wofür etwa $\log n$ Zeichen benötigt werden.

Mehrband-TM



Aus früheren VLs: Zu jeder Mehrband-TM existiert eine Ein-Band-TM

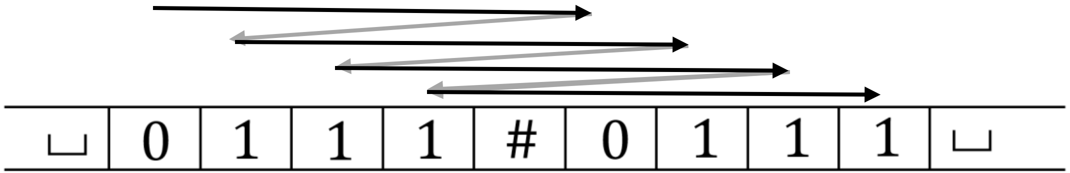
Reicht es, sich generell auf Ein-Band-TMs zu konzentrieren?

Beispiel 7.18

Sei COPY $= \{\, w\#w \mid w \in \{0{,}1\}^* \,\}$.

Zeige:

1. $\text{COPY} \in \text{DTIME}_1\big(O(n^2)\big)$
2. $\text{COPY} \in \text{DTIME}_2\big(O(n)\big)$
3. $\text{COPY} \notin \text{DTIME}_1\big(O(n)\big)$

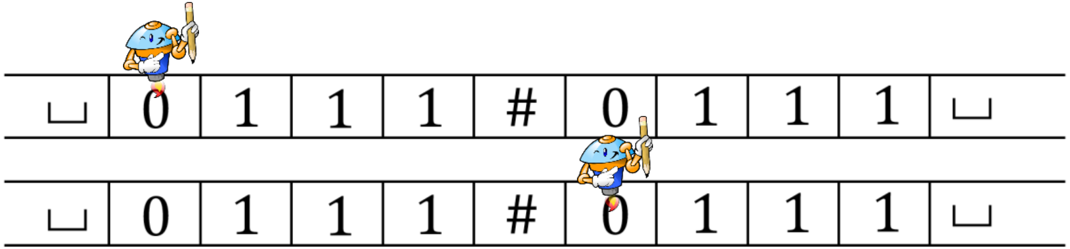


Zu 1.: Klar.
Vergleiche Position i des ersten Wortes mit Position i des zweiten Wortes.
Dafür muss man $O(n)$ Schritte durchführen.
Für $O(n)$ Symbole ergibt das eine Laufzeit von $O(n^2)$.

Beispiel 7.18

Sei $\text{COPY} = \{\, w\#w \mid w \in \{0,1\}^* \,\}$.
Zeige:

1. $\text{COPY} \in \text{DTIME}_1\big(O(n^2)\big)$
2. $\text{COPY} \in \text{DTIME}_2\big(O(n)\big)$
3. $\text{COPY} \notin \text{DTIME}_1\big(O(n)\big)$



Zu 2.: Auch recht klar.
Kopiere den Inhalt auf Band zwei und lese auf Band 1 die erste Kopie und auf Band 2 die zweite Kopie von w
Für $O(n)$ Symbole in w ergibt das eine Laufzeit von $O(n)$.

Beispiel 7.18

Sei $\mathrm{COPY} = \{\, w\#w \mid w \in \{0,1\}^* \,\}$.
Zeige:

1. $\mathrm{COPY} \in \mathrm{DTIME}_1\big(O(n^2)\big)$
2. $\mathrm{COPY} \in \mathrm{DTIME}_2\big(O(n)\big)$
3. $\mathrm{COPY} \notin \mathrm{DTIME}_1\big(O(n)\big)$

Zu 3.: Nicht so einfach zu sehen und lassen wir an dieser Stelle aus.
Man kann den Beweis über *Crossing-Sequenzen* führen.

Ein kleiner Wechsel des Berechnungsmodells kann also die Klasse der in Linearzeit lösbaren Probleme ändern! Die Klassen sind also nicht robust.

Moment mal

Das heißt, die ganzen Klassen können wir so nicht in der Praxis verwenden?!

Üblicherweise können wir annehmen, dass der Wechsel des Berechnungsmodells höchstens einen polynomiellen Blowup erzeugt:
Aus $f(n)$ wird $f(n^k)$

Robuste Komplexitätsklassen

Definition 7.19

$$\mathrm{L} = \mathrm{DSPACE}(O(\log n))$$
$$\mathrm{NL} = \mathrm{NSPACE}(O(\log n))$$
$$\mathrm{P} = \bigcup_{k\in\mathbb{N}} \mathrm{DTIME}\left(O\left(n^k\right)\right)$$
$$\mathrm{NP} = \bigcup_{k\in\mathbb{N}} \mathrm{NTIME}\left(O\left(n^k\right)\right)$$
$$\mathrm{PSPACE} = \bigcup_{k\in\mathbb{N}} \mathrm{DSPACE}\left(O\left(n^k\right)\right)$$
$$\mathrm{NPSPACE} = \bigcup_{k\in\mathbb{N}} \mathrm{NSPACE}\left(O\left(n^k\right)\right)$$

$$\mathrm{EXP} = \bigcup_{k\in\mathbb{N}} \mathrm{DTIME}\left(2^{O\left(n^k\right)}\right)$$
$$\mathrm{NEXP} = \bigcup_{k\in\mathbb{N}} \mathrm{NTIME}\left(2^{O\left(n^k\right)}\right)$$
$$\mathrm{EXPSPACE} = \bigcup_{k\in\mathbb{N}} \mathrm{DSPACE}\left(2^{O\left(n^k\right)}\right)$$
$$\mathrm{NEXPSPACE} = \bigcup_{k\in\mathbb{N}} \mathrm{NSPACE}\left(2^{O\left(n^k\right)}\right)$$

Natürlich kann man noch größere Klassen betrachten, wie $2\mathrm{EXP} = \bigcup_{k\in\mathbb{N}} \mathrm{DSPACE}\left(2^{2^{O\left(n^k\right)}}\right)$

Nächstes Mal