
4. Bandreduktionssätze

Wir untersuchen hier die Frage, inwieweit die Effizienz einer Turingmaschine von der Anzahl ihrer Bänder abhängt. Mit Hilfe einer einfachen, die Spurentechnik benutzenden Simulation zeigen wir, dass sich beim Übergang von Mehrband- zu Einbandmaschinen der Platzbedarf nicht erhöht, die Zeit i.a. aber quadratisch anwächst. Anhand eines Beispiels beweisen wir dann, dass dieser Zeitverlust i.a. unvermeidbar ist. Bei der Reduktion auf 2 Bänder können wir den Zeitverlust jedoch drastisch verkleinern. Hier erhöht sich die Laufzeit t nur auf $t \cdot \log(t)$. Ob der Zeitverlust auch in diesem Fall notwendigerweise auftritt, ist unbekannt.

4.1 SATZ. (1. BANDREDUKTIONSSATZ) *Zu jedem $t(n)$ -zeit- und $s(n)$ -platzbeschränkten k -Band-Turingakzeptor M ($k \geq 2$) kann man effektiv einen äquivalenten 1-Band-Turingakzeptor M' des ansonsten gleichen Typs angeben, der $O(t(n) \cdot \log(t(n)))$ -zeit- und $s(n)$ -platzbeschränkt ist.*

BEWEISIDEE. Das Arbeitsband des Akzeptors M' besteht aus $2k$ Spuren, je zwei Spuren für jedes der k Arbeitsbänder. Hierbei wird in der oberen Spur das M -Arbeitsfeld auf diesem Band durch ein $+$ (sonst $-$) markiert, während die untere Spur die Bandinschrift trägt:

-	-	-	+	-	-	-
b_{-3}	b_{-2}	b_{-1}	b_0	b_1	b_2	b_3

D.h. das Bandalphabet Γ' von M' ergibt sich aus dem Bandalphabet Γ von M durch

$$\Gamma' = \Gamma \cup (\{+, -\} \times \Gamma)^k.$$

In der *Initialisierungsphase* transformiert M' seine Startkonfiguration bei einer Eingabe w (der Länge n) in die Spurendarstellung der korrespondierenden Startkonfiguration von M . Im Falle einer off-line Maschine M muss hierzu nur das aktuelle Arbeitsfeld konvertiert werden (1 Schritt, kein Platz), während bei einer on-line Maschine M die Felder mit der Eingabe umgeschrieben werden müssen ($2n + 2 = O(n)$ Zeit, n Platz). In der Simulationsphase wird dann jeder M -Schritt durch eine M' -Schrittfolge wie folgt simuliert, wobei zu Beginn und Ende jeder M' -Schrittfolge das M' -Arbeitsfeld die Markierung des am weitesten links stehenden M -Arbeitsfeldes enthält: M' läuft zur Simulation des M -Schrittes zunächst nach rechts auf das am weitesten rechts stehende M -Arbeitsfeld und merkt sich hierbei in seinem Zustand neben dem M -Zustand

die gelesenen Inschriften der k M -Arbeitsfelder. Der Zustand von M' enthält dann den Bedingungsteil der auszuführenden M -Instruktion (oder der zulässigen Instruktionen bei nd. M). Im nächsten Schritt wird der Bedingungsteil durch den Operationsteil dieser M -Instruktion (oder einer der zulässigen Instruktionen bei nd. M) ersetzt. Beim Zurücklaufen werden dann die vorgeschriebenen M -Operationen ausgeführt.

Die Länge dieses M' -Zyklus zur Simulation eines M -Schrittes hängt von der aktuellen maximalen Distanz d zwischen zwei M -Arbeitsfeldern ab und ist linear in derselben. Da M $t(n)$ -zeitbeschränkt ist, muss bei Eingaben der Länge n jedoch stets $d \leq 2 \cdot t(n)$ gelten, weshalb der M' -Zyklus linear in $t(n)$ beschränkt ist.

Da höchstens $t(n)$ Zyklen erforderlich sind und $n < t(n)$ gilt, ergibt sich hieraus die gewünschte Zeitschranke $t'(n)$ mit

$$\begin{aligned} t'(n) &\leq O(n) + t(n) \cdot O(t(n)) \\ &\leq (t(n) + 1)(O(t(n))) = O(t(n)^2). \end{aligned}$$

Da M' nur den von M ebenfalls benutzten Platz benutzt, überträgt sich die Platzschranke $s(n)$ für M auf M' . \square

4.2 KOROLLAR. (i) $(N,D)\text{TIME}(t(n)) \subseteq (N,D)\text{TIME}_1(t(n) \cdot t(n))$

(ii) $(N,D)\text{SPACE}(s(n)) = (N,D)\text{SPACE}_1(s(n))$.

Um zu zeigen, dass bei der Reduktion von 2 Bändern auf 1 Band ein Zeitverlust quadratischer Ordnung i.a. unvermeidbar ist, betrachten wir die Sprache

$$L_{Sp} = \{w * w^R : w \in \Sigma_2^*\},$$

wobei w^R das Spiegelwort von w bezeichnet ($(w(0) \dots w(n))^R = w(n) \dots w(0)$).

Mit einer (deterministischen) 2-Band-Maschine lässt sich diese Sprache in Linearzeit erkennen.

4.3 LEMMA. $L_{Sp} \in \text{DTIME}_2^{(3)}(n+2)$.

BEWEIS. Ein deterministischer on-line 2-Band-Akzeptor M für L_{Sp} arbeitet wie folgt: M liest die Eingabe x unter gleichzeitigem Kopieren auf das zweite Band bis zum ersten Vorkommen eines $*$. Danach vergleicht M den Rest der Eingabe Buchstabe für Buchstabe mit dem Spiegelwort des auf Band 2 geschriebenen Anfangsstücks der Eingabe (indem M den Kopf auf Band 2 zurücklaufen lässt) und akzeptiert bei Gleichheit. \square

Die Möglichkeit – wie im Beweis oben – zwei Teilwörter an verschiedenen Stellen einer Eingabe in Linearzeit zu vergleichen, gibt einem eine 1-Band-TM i.a. nicht. Der naive Algorithmus hier, um L_{Sp} zu erkennen, ist, den ersten mit dem letzten Buchstaben zu vergleichen, den zweiten mit dem vorletzten Buchstaben etc., wobei jeder dieser Vergleiche (im Durchschnitt) linearen Zeitaufwand erfordert. Das Verfahren ist insgesamt also nur $O(n^2)$ -zeitbeschränkt.

Man kann das gerade beschriebene Verfahren zwar etwas optimieren, indem man z.B. statt einzelner Buchstaben Blöcke (fester Länge) vergleicht (und sich damit einige Wege spart), aber hierdurch wird nur eine Beschleunigung um einen konstanten Faktor erreicht. Obwohl es intuitiv klar zu sein scheint, dass sich eine 1-Band-TM, die L_{Sp} erkennt, im Wesentlichen wie beschrieben verhalten muss und daher (auf den

Eingaben ungerader Länge; die anderen können natürlich schnell verworfen werden) einen der Größenordnung nach quadratischen Zeitbedarf hat, ist es vielleicht nicht unbedingt klar, wie man diese untere Schranke beweisen soll. Wie schon früher bemerkt ist es in vielen Fällen bislang nicht gelungen, zwingend erscheinende untere Schranken tatsächlich zu beweisen. In diesem Fall liefern jedoch die sog. Crossing Sequences ein geeignetes Hilfsmittel, die gewünschte untere Schranke zu beweisen.

4.4 SATZ. Sei $t : \mathbb{N} \rightarrow \mathbb{N}$ rekursiv, sodass $L_{Sp} \in (N, D) \text{TIME}_1(t(n))$. Dann gilt $(2n+1)^2 \in O(t(2n+1))$.

BEWEIS. Sei M eine totale on-line 1-Band-Turingmaschine, die L_{Sp} erkennt. Wir haben zu zeigen, dass es eine Konstante $c \geq 1$ gibt, sodass

$$\forall n \exists w \in \Sigma_2^n (\text{time}_M(w * w^R) \geq \frac{(2n+1)^2}{c}) \quad (4.1)$$

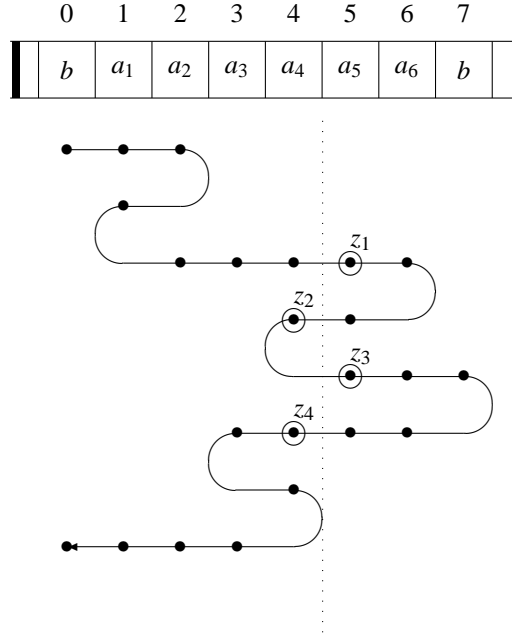
gilt. Hierzu gehen wir zunächst von einer deterministischen Turingmaschine M aus, über die wir noch folgende Annahmen machen: M ist eine Halbband-TM, M bewegt sich in jedem Schritt (um Stehenbleiben zu simulieren, gehen wir erst nach links und dann in einem Zusatzschritt nach rechts zurück) und M stoppt stets auf Feld 0 (stoppt M an anderer Stelle, so laufen wir vor dem "endgültigen" Stopp auf das linke Bandende zurück). Diese Zusatzannahmen sind zulässig, da der Übergang von einem beliebigen M zu einem äquivalenten derart normierten M' die Laufzeit nur um einen für die Gültigkeit von (4.1) belanglosen linearen Faktor erhöht. Z sei die Zustandsmenge von M und $c_0 = ||Z||$ die Anzahl der Zustände.

Zum Nachweis von (4.1) betrachten wir die Übergangsfolgen (Crossing Sequences) in akzeptierenden Rechnungen von M . Hierbei besteht die *Übergangsfolge*

$$\text{CS}_M(x, i) = \langle z_{i_1}, \dots, z_{i_i} \rangle$$

von M bei Eingabe x an der Stelle $i \geq 0$ aus den Zuständen, in denen sich die M -Rechnung bei Eingabe x nacheinander befindet, unmittelbar nachdem der LS-Kopf die Grenze zwischen den Feldern i und $i+1$ überschritten hat, also von i rechts nach $i+1$ oder umgekehrt von $i+1$ links nach i versetzt wurde.

Schematisch können wir die Übergangsfolge $\text{CS}_M(x, i)$ wie folgt darstellen: Da der Kopf von M nie stehenbleibt und da jede Rechnung auf Feld 0 beginnt und endet, können wir die Konfigurationsfolge der Rechnung als Punktfolge darstellen, wobei wir eine Konfiguration unter das zugehörige Arbeitsfeld schreiben können:



Das Diagramm veranschaulicht eine denkbare M -Rechnung bei Eingabe $x = a_1 \dots a_6$. Hierbei sind die nach Übergängen von Feld 4 nach Feld 5 bzw. umgekehrt erreichten Konfigurationen hervorgehoben und mit ihren Zuständen markiert. Es ist also hier $\text{CS}_M(x, 4) = \langle z_1, z_2, z_3, z_4 \rangle$. Die zugehörigen Bewegungen sind hierbei alternierend nach rechts und links. Da die Rechnung auf Feld 0 beginnt und endet, ist die erste Bewegung eine Linksrechtsbewegung und die Länge $L(\gamma)$ jeder Übergangsfolge γ gerade.

Wichtig ist zu beobachten, dass die zu Übergangsfolgen an verschiedenen Stellen gehörenden Zeitpunkte der M -Rechnung paarweise verschieden sind, sich die Rechnungslänge also gerade als Summe der Längen der Übergangsfolgen ergibt:

$$\text{time}_M(x) = \sum_{i=0}^{\infty} L(\text{CS}_M(x, i)). \quad (4.2)$$

Zum Nachweis von (4.1) genügt es daher zu zeigen, dass man zu gegebenem (hinreichend großem) n ein geeignetes Wort $w_n \in \Sigma_2^n$ finden kann, sodass die zur M -Rechnung bei Eingabe $w_n * w_n^R$ gehörenden Übergangsfolgen hinreichend lang sind. Um dies zu präzisieren, definieren wir die durchschnittliche Länge

$$l(n, i) = \frac{\sum_{w \in \Sigma_2^n} (L(\text{CS}_M(w * w^R, i)))}{|\Sigma_2^n|}$$

der Übergangsfolgen von M an der Stelle i für korrekte Eingaben der Länge $2n + 1$. Wegen (4.2) gibt es dann ein Wort $w_n \in \Sigma_2^n$, sodass

$$\text{time}_M(w_n * w_n^R) \geq \sum_{i=0}^{n-1} l(n, i)$$

gilt. Können wir daher Konstanten $d_1, d_2 \geq 1$ finden, sodass

$$\forall n \forall i (d_2 \leq i < n \Rightarrow l(n, i) \geq \frac{i}{d_1}) \quad (4.3)$$

gilt, so erhalten wir hiermit

$$\text{time}_M(w_n * w_n^R) \geq \sum_{i=d_2}^{n-1} \frac{i}{d_1} \geq \frac{n^2}{4d_1} \geq \frac{(2n+1)^2}{36d_1}$$

für fast alle n und damit (4.1) für $c = 36d_1$.

Im Rest des Beweises zeigen wir (4.3) (für geeignetes d_1, d_2). Geleitet werden wir hierbei von folgender Intuition: Um eine korrekte Eingabe $w * w^R$ zu überprüfen, muss M zwischen dem linken und rechten Wortteil hin- und herlaufen, um die Spiegeigenschaften lokal zu verifizieren. Dabei muss sich M die an einer Stelle in dem einen Teil gesehenen Buchstaben im Zustand merken, um diese mit der Inschrift der entsprechenden Stelle im anderen Teil zu vergleichen. Da die Anzahl der M -Zustände konstant ist, kann sich M hierbei immer nur Information über einen konstant großen Teil von w bzw. w^R merken. Ist daher an einer Stelle $i < n$ die Übergangsfolge im Schnitt zu klein, so kann über die Grenze i nicht die vollständige Information über den Wortanfang $a_1 \dots a_i$ von w an das Wortende w^R (bzw. umgekehrt) transportiert worden sein, um diese Wortteile vollständig zu vergleichen. Gewisse Defekte in der Symmetrie der Eingabe entgehen also M und M wird (im Widerspruch zur Annahme, dass $L(M) = L_{Sp}$ gilt) auch fehlerhafte Eingaben akzeptieren.

Um dies zu präzisieren, benutzen wir folgende fundamentale Eigenschaft der Übergangsfolgen (für beliebiges M).

Behauptung 1. Seien $x = a_1 \dots a_m$ und $\hat{x} = \hat{a}_1 \dots \hat{a}_m$ von M akzeptierte Wörter, sei $1 \leq i < m$ und gelte $\text{CS}_M(x, i) = \text{CS}_M(\hat{x}, i)$. Dann akzeptiert M auch das Wort $\tilde{x} = a_1 \dots a_i \hat{a}_{i+1} \dots \hat{a}_m$.

BEWEIS. Seien $\alpha_1, \dots, \alpha_l$ die M -Rechnung bei Eingabe x , $\alpha_{i_1}, \dots, \alpha_{i_k}$ ($i_1 < i_2 < \dots < i_k$) die Konfiguration unmittelbar nach Überschreiten der Grenze zwischen den Feldern i und $i+1$ (in beide Richtungen) in der Rechnung, und seien z_{i_1}, \dots, z_{i_k} die zugehörigen Zustände, d.h. $\text{CS}_M(x, i) = \langle z_{i_1}, \dots, z_{i_k} \rangle$. Analog seien $\hat{\alpha}_1, \dots, \hat{\alpha}_l$ sowie $\hat{\alpha}_{j_1}, \dots, \hat{\alpha}_{j_k}$ ($j_1 < j_2 < \dots < j_k$) und $\hat{z}_{j_1}, \dots, \hat{z}_{j_k}$ passend zur Eingabe \hat{x} gewählt, also $\text{CS}_M(\hat{x}, i) = \langle \hat{z}_{j_1}, \dots, \hat{z}_{j_k} \rangle$, wobei nach Annahme $z_{i_p} = \hat{z}_{j_p}$ für $1 \leq p \leq k$ gilt. Bei Eingabe $\tilde{x} = a_1 \dots a_i \hat{a}_{i+1} \dots \hat{a}_m$ verhält sich M dann wie bei Eingabe x so lange das Arbeitsfeld eins der Felder $0, \dots, i$ ist (linker Bandteil) und wie bei Eingabe \hat{x} , wenn das Arbeitsfeld Adresse $\geq i+1$ hat (rechter Bandteil). Die Rechnung lässt sich also als die Konfigurationsfolge

$$\beta_1, \dots, \beta_{i_1}, \hat{\beta}_{j_1+1}, \dots, \hat{\beta}_{j_2-1}, \beta_{i_2}, \dots, \dots, \hat{\beta}_{j_k-1}, \beta_{i_k}, \dots, \beta_l.$$

beschreiben, wobei für $i_{2s} \leq m < i_{2s+1}$ und $0 \leq s < k/2$ (wobei $i_0 := j_0 := 1$)

$$\begin{aligned} l(\beta_m) &= l(\alpha_m) \\ z(\beta_m) &= z(\alpha_m) \\ r(\beta_m) &= r(\hat{\alpha}_{j_{2s}}) \end{aligned}$$

und für $j_{2s+1} \leq m < j_{2s+2}$ und $0 \leq s < k/2$

$$\begin{aligned} l(\hat{\beta}_m) &= l(\alpha_{i_{2s+1}}) \\ z(\hat{\beta}_m) &= z(\hat{\alpha}_m) \\ r(\hat{\beta}_m) &= r(\hat{\alpha}_m). \end{aligned}$$

Hierbei sind $l(\beta), r(\beta), z(\beta)$ die Inschrift des linken Bandteiles, die Inschrift des rechten Bandteiles und der Zustand von M bei Konfiguration β .

Dass die Übergänge “passen”, liegt an der Gleichheit der Übergangsfolgen für die Eingaben x und \hat{x} an der Stelle i . Da sich die Stoppkonfiguration bei Eingabe \tilde{x} von der Stoppkonfiguration bei Eingabe x auf den Feldern $[0, \dots, i]$ nicht unterscheidet, gilt wegen $x \in L(M)$ auch $\tilde{x} \in L(M)$.

Für das speziell gegebene M ergibt sich aus Behauptung 1:

Behauptung 2. Seien $w = a_1 \dots a_n, \hat{w} = \hat{a}_1 \dots \hat{a}_n \in \Sigma_2^n$, sei $1 \leq i \leq n$ und gelte $a_1 \dots a_i \neq \hat{a}_1 \dots \hat{a}_i$. Dann sind $\text{CS}_M(w * w^R, i)$ und $\text{CS}_M(\hat{w} * \hat{w}^R, i)$ verschieden.

BEWEIS. Würden die Übergangsfolgen übereinstimmen, so würde nach Behauptung 1

$$y = a_1 \dots a_i \hat{a}_{i+1} \dots \hat{a}_n * \hat{a}_n \dots \hat{a}_{i+1} \hat{a}_i \dots \hat{a}_1 \in L(M)$$

gelten. Wegen $a_1 \dots a_i \neq \hat{a}_1 \dots \hat{a}_i$ gilt jedoch $(a_1 \dots a_i)^R \neq \hat{a}_i \dots \hat{a}_1$. Da $(y_1 y_2)^R = y_2^R y_1^R$ für Wörter y_1, y_2 gilt, folgt hieraus aber, dass y nicht die Gestalt $y = \tilde{w} * \tilde{w}^R$ hat. Es wäre also $L(M) \neq L_{Sp}$ im Widerspruch zur Annahme.

Um hiermit die gewünschte Abschätzung (4.3) der durchschnittlichen Übergangsfolgenlängen zu erhalten, fassen wir die Binärwörter w der Länge n zusammen, deren zugehörigen Elemente $w * w^R$ von L_{Sp} an einer Stelle i zu einer vorgegebenen Übergangsfolge γ bzw. zu einer Folge, deren Länge durch einen Wert l beschränkt ist, führen:

$$\begin{aligned} A_{n,i,\gamma} &= \{w \in \Sigma_2^n : \text{CS}_M(w * w^R, i) = \gamma\} \\ A_{n,i,l} &= \{w \in \Sigma_2^n : L(\text{CS}_M(w * w^R, i)) \leq l\} \\ &= \bigcup \{A_{n,i,\gamma} : L(\gamma) \leq l\}. \end{aligned}$$

Da offensichtlich zumindest für die Hälfte der Wörter $w \in \Sigma_2^n$ die zugehörige Übergangsfolge $\text{CS}_M(w * w^R, i)$ eine Länge unterhalb der zweifachen durchschnittlichen Länge $l(n, i)$ haben muss, gilt

$$|A_{n,i,2l(n,i)}| \geq \frac{1}{2} |\Sigma_2^n| = \frac{1}{2} \cdot 2^n = 2^{n-1}.$$

Da weiter die Anzahl der verschiedenen Übergangsfolgen der Länge $\leq 2l(n, i)$ durch

$$c_0^0 + c_0^2 + \dots + c_0^{2l(n,i)} \leq c_0^{2l(n,i)+1}$$

beschränkt ist (c_0 ist die Anzahl der M -Zustände), muss es daher eine Folge γ der Länge $\leq 2l(n, i)$ geben, sodass

$$|A_{n,i,\gamma}| \geq \frac{|A_{n,i,2l(n,i)}|}{c_0^{2l(n,i)+1}} = \frac{2^{n-1}}{c_0^{2l(n,i)+1}}$$

gilt. Andererseits lässt sich wegen Behauptung 2 die Anzahl der Wörter w , die zu derselben Übergangsfolge γ an einer Stelle i führen, durch

$$|A_{n,i,\gamma}| \leq 2^{n-i}$$

nach oben abschätzen. Würde nämlich diese Schranke nach oben überschritten, so müssten sich zwei Wörter $w, \hat{w} \in A_{n,i,\gamma}$ in den ersten i Buchstaben unterscheiden, da für festes $a_1 \dots a_i \in \Sigma_2^i$

$$|\{w \in \Sigma_2^n : w(0) \dots w(i-1) = a_1 \dots a_i\}| = 2^{n-i}.$$

Dies ist jedoch nach Definition von $A_{n,i,\gamma}$ und nach Behauptung 2 nicht möglich.

Aus den Schranken für $|A_{n,i,\gamma}|$ folgt nun

$$\frac{2^{n-1}}{c_0^{2l(n,i)+1}} \leq 2^{n-i}.$$

Auflösen dieser Ungleichung nach $l(n,i)$ liefert

$$\begin{aligned} c_0^{2l(n,i)+1} &\geq \frac{2^{n-1}}{2^{n-i}} = 2^{i-1} \\ \log(c_0)(2l(n,i)+1) &\geq i-1 \\ l(n,i) &\geq \frac{i - (2\log(c_0) + 1)}{2\log(c_0)} \end{aligned}$$

woraus (4.3) für $d_1 = d_2 = 4\log(c_0) + 2$ folgt.

Für nd. M argumentiert man wie oben, wobei man für jede Eingabe $w * w^R$ eine feste (z.B. die "kleinste") akzeptierende M -Rechnung der Analyse zugrunde legt. \square

4.5 KOROLLAR. Sei t eine rekursive Funktion für die $t(n) <_{f.ü.} \frac{1}{c}n^2$ für fast alle $c \geq 1$ gilt. Dann gilt

$$\text{DTIME}_2^{(3)}(O(n)) \not\subseteq \text{NTIME}_1^{(3)}(t(n)).$$

BEWEIS. Nach Lemma 4.3 und Satz 4.4 gilt

$$L_{Sp} \in \text{DTIME}_2^{(3)}(O(n)) - \text{NTIME}_1^{(3)}(t(n)).$$

\square

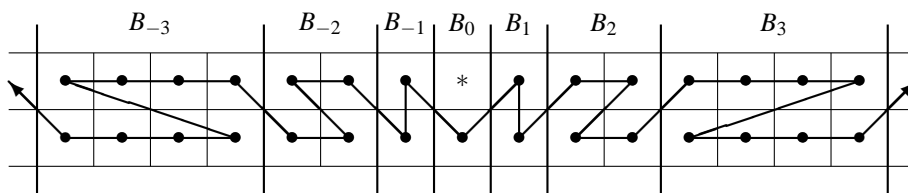
4.6 BEMERKUNG. Korollar 4.5 gilt auch für das binäre Alphabet und es genügt $t(n) < \frac{1}{c}n^2$ für unendliche viele n zu fordern. Um dies zu zeigen, kann man z.B. die Sprache $L_P = \{w \in \Sigma_2^* : w = w^R\}$ der Palindrome über Σ_2 betrachten. Mit einer 2-Band-TM lässt sich die Palindromeigenschaft in linearer Zeit erkennen, d.h. $L_P \in \text{DTIME}(O(n))$. Dass für einen $t(n)$ -zeitbeschränkten 1-Band-Turingakzeptor M , der L_P erkennt, $n^2 \in O(t(n))$ gilt, lässt sich aus Satz 4.4 ableiten. (Gäbe es ein M , das diese quadratische Zeitschranke unendlich oft unterschreitet, so könnte man hieraus einen Akzeptor M' für L_{Sp} gewinnen, der die Schranke in 4.4 unendlich oft unterschreitet; s.Übungen.)

Reduziert man auf 2 Bänder, so lässt sich der quadratische Zeitverlust vermeiden. Hier ist die simulierende Maschine nur um einen logarithmischen Faktor langsamer. Wegen des sehr langsamen Wachstums der Logarithmus-Funktion, kommt dies einem (nach dem Beschleunigungssatz belanglosen) linearen Faktor sehr nahe.

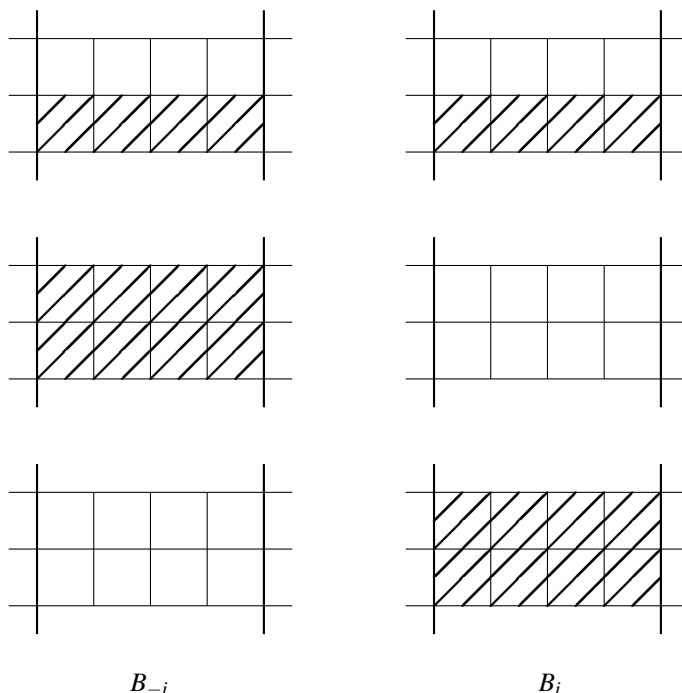
4.7 SATZ. (2. BANDREDUKTIONSSATZ) Zu jedem $t(n)$ -zeitbeschränkten k -Band online Turingakzeptor M ($k \geq 1$) kann man effektiv einen äquivalenten 2-Band-Turingakzeptor M' des ansonsten gleichen Typs angeben, der $O(t(n) \cdot \log(t(n)))$ -zeitbeschränkt ist.

BEWEIS. Ähnlich wie im Beweis des 1. Bandreduktionssatzes simuliert M' den Akzeptor M Schritt-für-Schritt unter Verwendung der Spurentechnik. Hierzu werden jedem M -Band zwei Spuren auf dem ersten M' -Band zugeordnet (die wiederum durch das Bandalphabet von M' realisiert werden). Das zweite Band benutzt M' nur als Hilfsband, um Speicheroperationen auf dem 1. Band (Kopieren, Abstandmessen) effizient durchzuführen. Neu ist, dass M' die Maschine M so interpretiert, dass diese nicht ihre Köpfe auf dem (festen) Band sondern das Band unter den (festen) Köpfen bewegt, also z.B. eine Linksbewegung des Kopfes durch eine Rechtsverschiebung der Daten (um ein Feld) realisiert. Damit hierbei nicht alle Daten auf dem Band verschoben werden müssen, benutzt M' eine Darstellung der M -Bänder mit Lücken (so dass die Daten hinter der ersten Lücke nicht bewegt werden müssen).

Da M' alle Bänder von M gleichbehandelt, beschreiben wir die Arbeitsweise von M' nur für ein festes M -Band. Die zwei dem M -Band zugeordneten Spuren dienen beide der Aufnahme der M -Bandinschrift und sind ausgehend von Feld 0 in Blöcke eingeteilt, deren Längen sich laufend verdoppeln. Block B_0 besteht aus Feld 0, enthält unten die aktuelle Inschrift des M -Arbeitsfeldes und oben das Symbol $*$ zur Kennzeichnung der Bandmitte. Die sich anschließenden Blöcke B_i ($i \neq 0$) haben die Länge $2^{|i|-1}$. Dabei liegen die Daten in der oberen Spur solch eines Blockes näher zur Bandmitte als die in der unteren Spur. Folgendes Diagramm veranschaulicht die Blockstruktur und die Reihenfolge, in der das M -Band gespeichert wird.

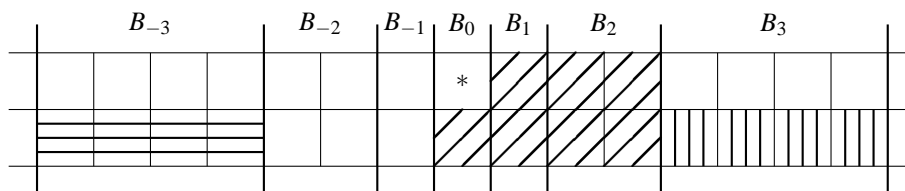


Die Blöcke werden jedoch nicht vollständig gefüllt, sondern vor und nach jedem Simulationszyklus eines M -Schrittes wird folgende *Blockbelegungsbedingung* (BBB) gelten, die sich auf die dualen Blöcke B_{-i} und B_i bezieht: Für jedes $i > 0$ sind entweder in B_i und B_{-i} die unteren Spuren belegt (*balancierte Belegung*) oder einer der Blöcke ist vollständig belegt und der andere Block ist vollständig frei:

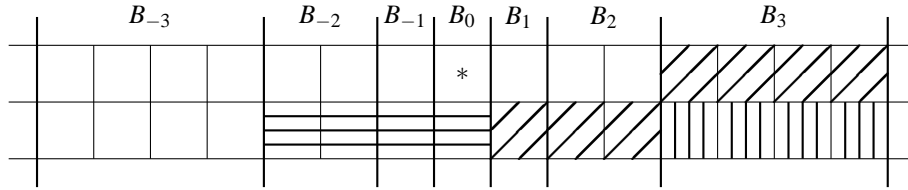


Im Block B_0 ist immer das untere Feld belegt und das obere Feld mit * beschriftet. Die Blockbelegung ist so gewählt, dass bei einer Rechts- (bzw. Links-)verschiebung nur die Blöcke bis (einschließlich) zum ersten nicht vollständig belegten Block $B_i (i > 0)$ und deren duale Blöcke $B_{-j} (1 \leq j \leq i)$ neu beschriftet werden müssen. Realisiert wird dabei die Rechtsverschiebung durch die folgende B_i -Operation (für das entsprechende i). Angewendet werden kann die B_i -Operation ($i > 0$) nur, falls B_1, \dots, B_{i-1} vollständig belegt sind, B_i nicht vollständig belegt ist, und BBB gilt (d.h. $B_{-1}, \dots, B_{-(i-1)}$ sind frei und auf B_{-i} und B_i trifft eine der beiden ersten oben schematisch dargestellten Situationen zu). Die B_i -Operation schreibt dann die Inhalte der Blöcke B_0, \dots, B_{i-1} in die unteren Spuren von B_1, \dots, B_{i-1} und in die untere Spur von B_i , falls diese frei ist, sonst in die obere Spur. Weiter wird die obere Spur von B_{-i} , falls diese belegt ist, sonst die untere Spur, in die unteren Spuren von $B_{-(i-1)}, \dots, B_0$ geschrieben (jeweils in der richtigen Reihenfolge). Linksverschiebungen werden durch symmetrisch definierte B_{-i} -Operationen realisiert.

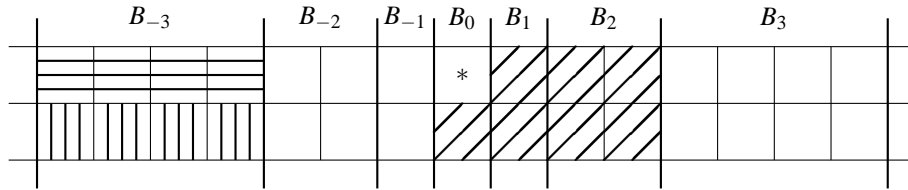
Für $i = 3$ lässt sich (in Abhängigkeit von der Belegung von B_i) die B_i -Operation wie folgt schematisch darstellen: Die Belegung



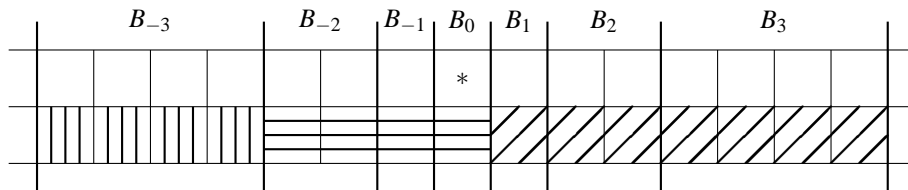
geht über in



und die Belegung



geht über in



Wichtig ist festzuhalten, dass nach einer Rechtsverschiebung vermöge einer B_i -Operation die BBB wieder hergestellt ist und die Blöcke B_1, \dots, B_{i-1} balanciert belegt sind. Um zu sehen, dass die verschobenen Daten an den gewünschten Stellen exakt Platz finden, beobachtet man den folgenden Zusammenhang zwischen der Länge $L(B_i)$ und der Größe $G(B_i)$ (Anzahl der unterzubringenden M -Buchstaben) für die verschiedenen Blöcke (wobei $i \geq 1$):

$$L(B_0) = G(B_0) = 2^0 = 1$$

$$L(B_{\pm i}) = 2^{i-1} \tag{4.4}$$

$$G(B_{\pm i}) = 2 \cdot L(B_{\pm i}) = 2^i$$

$$G(B_0) + \dots + G(B_{\pm(i-1)}) = \sum_{j=0}^{i-1} 2^j = 2^i - 1 \tag{4.5}$$

$$L(B_1) + \dots + L(B_i) = \sum_{j=1}^i 2^{j-1} = 2^i - 1 \tag{4.6}$$

Wir beschreiben M' nun genauer (jedoch nicht formal) um den Zeitbedarf von M' abschätzen zu können. Dabei betrachten wir zunächst die *Initialisierungsphase* und das *Anlegen der Blockstruktur*.

Die Eingabe wird in Spurendarstellung und gemäß der Blockstruktur kodiert, wobei die hierzu erforderlichen Blöcke zusammen mit den dualen Blöcken durch Markierung (mit Hilfe des Alphabets, z.B. auf einer zusätzlichen Spur) des jeweils äussersten Feldes eines Blocks kenntlich gemacht werden. Bei der erstmaligen Blockbelegung wird dabei stets die balancierte Belegung gewählt.

Bei Eingabe $a_1 \dots a_5$ wird also beispielsweise das Eingabeband in

B_{-3}				B_{-2}		B_{-1}	B_0	B_1	B_2			B_3			
							*								
b	b	b	b	b	b	b	b	a_1	a_2	a_3	a_4	a_5	b	b	

konvertiert und die zu den anderen Bändern gehörenden Spuren auf dem selben Abschnitt geschaffen, wobei die unteren Spuren dort durchgehend mit Blanks beschriftet sind.

Wird im Laufe der Rechnung der bereits konvertierte Bandteil verlassen, so wird – bevor weitergerechnet wird – zunächst der nächste Block und der dazu duale Block geschaffen und balanciert mit Blanks beschriftet.

Bei der Schaffung neuer Blöcke geht M' dabei wie folgt vor: Wird der konvertierte Bereich rechts (links: symmetrisch) – etwa am Ende von Block B_{i-1} – verlassen, so werden B_i und B_{-i} wie folgt installiert: Erreicht M' von links kommend einen noch nicht konvertierten Buchstaben, so erkennt es hieran, dass ein neuer Block geschaffen werden muss. Es merkt sich den Zustand (für die später fortzusetzende Rechnung) und läuft zur Bandmitte zurück, wobei es die zurückgelegte Wegstrecke W auf Band 2 markiert. Nach (4.4) und (4.6) ist die Länge von W gerade die Länge der neu zu schaffenden Blöcke B_i und B_{-i} . M' kann also als nächstes an das linke konvertierte Bandende laufen, dort B_{-i} installieren (gleichzeitiges Durchlaufen von W auf Band 2 gibt die Länge an), dann analog am rechten Ende B_i installieren und schließlich (unter gleichzeitigem Löschen von W auf Band 2) auf das erste Feld von B_i zurücklaufen, um die eigentlich Rechnung fortzusetzen.

Hierzu wird der Weg vom rechten Ende von B_i zum linken Ende von B_{-i} in jede Richtung einmal durchlaufen. Nach (4.4) und (4.6) erfordert das gerade $8 \cdot L(B_i)$ Schritte. Da nur Blöcke erzeugt werden, von denen M ein Feld benutzt, ist (wiederum wegen (4.4) und (4.6)) die Länge des konvertierten Bandes durch $4t(n)$ beschränkt. Es folgt, dass die Initialisierungsphase und der später während der Simulationsphase entstehende Zeitbedarf zur Schaffung der Blockstruktur linear in $t(n)$ beschränkt ist und daher bei der Analyse der gesamten Laufzeit vernachlässigt werden kann.

Um den Zeitbedarf der Simulationsphase zu bestimmen beobachten wir, dass M' zur Simulation eines M -Schrittes in einem Schritt Zustand und Bandinschriften aktualisieren kann. (Zu Beginn und Ende eines Simulationszyklus steht M' auf Feld 0, das alle M -Arbeitsfelder enthält.) Der Zeitbedarf wird also im Wesentlichen dadurch bestimmt, wieviel Zeit die Simulation der Bewegungen der einzelnen M -Bänder erfordert. Da die Anzahl der Bänder konstant und die Simulation von Rechts- und Linksverschiebungen symmetrisch ist, genügt es also eine Zeitschranke der Größenordnung $O(t(n) \cdot \log(t(n)))$ für die Simulation aller Rechtsverschiebungen auf einem festen Band nachzuweisen. Hierzu genügt es wiederum die benötigte Zeit für die zu den Rechtsverschiebungen zugehörigen B_i -Operationen entsprechend abzuschätzen.

D.h. zum Nachweis der gewünschten Zeitschranke für M' genügt es für

$$\begin{aligned} \#\beta_i(n) &= \text{maximale, theoretisch mögliche Anzahl von erforderlichen } B_i\text{-Operationen bei der Simulation von } M \text{ für eine} \\ &\quad \text{Eingabe der Länge } n \text{ (für ein festes } M\text{-Band)} \\ \tau\beta_i &= \text{erforderliche Zeit zur Ausführung einer } B_i\text{-Operation} \end{aligned}$$

zu zeigen, dass

$$\sum_{i=1}^{\infty} \#\beta_i(n) \cdot \tau\beta_i \in O(t(n) \cdot \log(t(n))) \quad (4.7)$$

gilt.

Zum Beweis von (4.7) erinnern wir uns zunächst, dass ein Block B_i nur dann geschaffen wird, wenn das erste Feld von B_i von M benutzt wird, also eine Adresse $\leq t(n)$ hat. Wegen (4.6) bedeutet dies, dass $2^i \leq t(n)$ also $i \leq \log(t(n))$ gilt. Für i mit $i > \log(t(n))$ gilt deshalb $\#\beta_i(n) = 0$ weshalb wir in (4.7) die obere Summationsgrenze beschränken können:

$$\sum_{i=1}^{\infty} \#\beta_i(n) \cdot \tau\beta_i \leq \sum_{i=1}^{\log(t(n))} \#\beta_i(n) \cdot \tau\beta_i.$$

Zur Abschätzung der verbliebenen $\#\beta_i(n)$ erinnern wir uns weiter, dass nach einer B_i -Operation die Blöcke B_1, \dots, B_{i-1} nur unten belegt sind. Eine weitere B_i -Operation wird aber erst dann wieder erforderlich, wenn diese Blöcke vollständig belegt sind. Hierzu ist aber die Simulation von

$$L(B_1) + \dots + L(B_{i-1}) = 2^{i-1} - 1$$

(vgl. (4.6)) Rechtsverschiebungen notwendig. Zwischen zwei B_i -Operationen liegen also mindestens ebensoviele M -Schritte, weshalb

$$\#\beta_i(n) \leq \frac{t(n)}{2^{i-1} - 1}$$

gilt. Schließlich können wir den Zeitaufwand $\tau\beta_i$ einer B_i -Operation linear in der Blocklänge $L(B_i) = 2^{i-1}$ abschätzen. Hierzu genügt es zu beobachten, dass wegen (4.4) und (4.6) die Gesamtlänge der Blöcke B_{-i}, \dots, B_i linear in $L(B_i)$ beschränkt ist und die bei der B_i -Operation erforderlichen Datenverschiebungen einen Zeitaufwand linear in der Datenlänge sowie der Distanz der Verschiebung (beides wiederum durch $L(B_i)$ linear beschränkt!) haben. Den Inhalt w der Blöcke B_0, \dots, B_{i-1} können wir nämlich in $\frac{3}{2}|w|$ Schritten in der richtigen Reihenfolge unter gleichzeitigem Löschen auf Band 2 kopieren und dann – wiederum unter gleichzeitigem Löschen und wiederum in $O(|w|)$ Schritten – zurück in die gewünschten Spuren der Blöcke B_1, \dots, B_i schreiben. Da $|w| \in O(L(B_i))$, genügt der erste Teil der B_i -Operation also der gewünschten Zeitschranke. Für den zweiten Teil zeigt man dies entsprechend.

Durch Einsetzen der nachgewiesenen Schranken für $\#\beta_i(n)$ und $\tau\beta_i$ erhalten wir

$$\begin{aligned}
\sum_{i=1}^{\log(t(n))} \#\beta_i(n) \cdot \tau\beta_i &\leq \sum_{i=1}^{\log(t(n))} \frac{t(n)}{2^{i-1}-1} \cdot O(2^i) \\
&\leq \sum_{i=1}^{\log(t(n))} O(t(n)) \\
&= O(t(n) \cdot \log(t(n)))
\end{aligned}$$

und damit den Nachweis von (4.7). Dies beendet den Beweis. \square

4.8 KOROLLAR. *Für jede rekursive Funktion t gilt*

$$(N,D)\text{TIME}(t(n)) \subseteq (N,D)\text{TIME}_2(O(t(n) \log(t(n)))).$$

Es ist unbekannt, ob sich der 2. Bandreduktionssatz verschärfen lässt. Ebensovienig wissen wir, ob durch Reduktion auf eine größere (konstante) Zahl von Bändern eine schnellere Simulation möglich ist.