
7. Deterministische Platz- und Zeithierarchiesätze

Wir untersuchen hier die Frage, wie man zu einer gegebenen Platzschranke (oder Zeitschranke) $s(n)$ eine größere Schranke $S(n)$ finden kann, die die Lösung zusätzlicher Probleme ermöglicht. Notwendige Anforderungen an solch eine Schranke ergeben sich aus der Linearen Kompression und dem allgemeinen Lückensatz. Aus ersterem ergibt sich die Notwendigkeit, dass $S(n) \notin O(s(n))$ gelten muss. Aus letzterem folgt, dass es nicht genügt, $s(n)$ um einen vorgegebenen rekursiven Wachstumsfaktor $f(n)$ zu $S(n) = f(s(n))$ zu vergrößern: Um die Existenz einer Sprache $A \in \text{DSPACE}(S(n)) - \text{DSPACE}(s(n))$ zu sichern, müssen wir explizit fordern, dass eine Platzkostenfunktion $\sigma(n)$ zwischen $s(n)$ und $S(n)$ liegt. Beide Anforderungen zusammen erfordern also die Existenz einer Kostenfunktion $\sigma(n)$, die $\sigma(n) \notin O(s(n))$ und $\sigma(n) \leq S(n)$ für fast alle n erfüllt. Hinzu kommt, dass wir Lücken bei der Zeit- und Platzkomplexität für sehr kleine Schranken beobachtet haben, weshalb die Schranke $s(n)$ nicht zu klein sein darf.

Im Falle der deterministischen Platzkomplexität sind diese notwendigen Anforderungen an die Schranke $S(n)$ im Wesentlichen auch hinreichend, während bei der deterministischen Zeitkomplexität die bislang bekannten notwendigen und hinreichenden Bedingungen nicht zusammenfallen. Dieser Unterschied erklärt sich dadurch, dass die Beweise der Hierarchiesätze die Bandreduktion benutzen, die beim Platzbedarf verlustfrei durchführbar ist aber bei der Rechenzeit die Erhöhung einer Schranke $t(n)$ auf $t(n) \cdot \log(t(n))$ bewirkt (wenn wir die effizientere Reduktion auf 2 Bänder betrachten).

Statt Kostenfunktionen betrachten wir im folgenden sogenannte konstruierbare Funktionen. Diese treten als Kostenfunktionen von Maschinen auf, deren Kosten von der Eingabelänge, aber nicht von den speziellen Eingaben abhängen.

7.1 DEFINITION. Eine total rekursive Funktion $s : \mathbb{N} \rightarrow \mathbb{N}$ ist *platzkonstruierbar*, wenn es einen deterministischen off-line Turingakzeptor M über dem Eingabealphabet Σ_2 gibt, dessen Platzbedarf bei jeder Eingabe der Länge n gerade $s(n)$ ist. Entsprechend ist die total rekursive Funktion $t : \mathbb{N} \rightarrow \mathbb{N}$ *zeitkonstruierbar*, wenn es einen deterministischen on-line Turingakzeptor M' gibt, dessen Rechenzeit bei jeder Eingabe der Länge n gerade $t(n)$ ist.

Die üblichen monotonen Funktionen, die man in der Arithmetik betrachtet, wie z.B. der Logarithmus $\log(n)$, die Wurzel \sqrt{n} , Polynome, die Exponentialfunktion 2^n und die Fakultät $n!$, sind platzkonstruierbar und – bis auf die sublinearen Funktionen – auch zeitkonstruierbar. (Wir verzichten hier auf den Beweis dieser Feststellung.) Darüberhinaus kann man beliebig schnell (rekursiv) wachsende konstruierbare Funktionen angeben:

7.2 LEMMA. Zu jeder total rekursiven Funktion $f : \mathbb{N} \rightarrow \mathbb{N}$ gibt es eine streng monotone, zeit(platz)-konstruierbare Funktion f' mit $f \leq f'$.

BEWEIS. Sei M eine det. Turingmaschine, die f bezüglich der Unärdarstellung berechnet. Da schon das Aufschreiben der Ausgabe $0^{f(n)}$ $f(n)$ Schritte benötigt, gilt dann $\text{time}(0^n) \geq f(n)$. Man definiert nun einen Akzeptor M' mit binärem Eingabealphabet Σ_2 , der bei einer Eingabe w deren Länge n unär als 0^n darstellt und dann sukzessive M für die Eingaben $0^0, 0^1, \dots, 0^n$ simuliert. Da die Rechenzeit von M' nur von der Eingabengänge abhängt, ist $f'(n) := \text{time}_{M'}(0^n)$ zeitkonstruierbar. Nach Definition von M' ist aber f' streng monoton und $f' \geq f$. (Im Fall der Platzkonstruierbarkeit betrachtet man $f'(n) := \text{space}_{M'}(0^n)$, wobei man zusätzlich annimmt, dass die Maschine M ihre Ausgabe zusätzlich auf ein Arbeitsband schreibt, wodurch der Platzbedarf bei mindestens $f(n)$ liegt.) \square

Mi Hilfe des Konzepts der Konstruierbarkeit können wir nun die Hierarchiesätze formulieren. Wir beginnen mit dem Platzhierarchiesatz.

7.3 SATZ. (PLATZHIERARCHIESATZ) Seien $s(n), S(n) \geq \log(n)$ rekursive Funktionen, für die

$$(S1) \ S(n) \notin O(s(n)), \text{ d.h. } \liminf_{n \rightarrow \infty} \frac{s(n)}{S(n)} = 0,$$

(S2) $S(n)$ platzkonstruierbar

gilt. Dann gilt $\text{DSPACE}(S(n)) \not\subseteq \text{DSPACE}(s(n))$. Werden $s(n)$ und $S(n)$ zusätzlich so gewählt, dass $s(n) \leq S(n)$ für fast alle n gilt, so gilt also

$$\text{DSPACE}(s(n)) \subsetneq \text{DSPACE}(S(n)).$$

7.4 BEISPIEL. Der Platzhierarchiesatz impliziert z.B. die Echtheit der folgenden Inklusionen mit polynomiellen und exponentiellen Schrankenfunktionen ($(k \geq 0, l \geq 1)$):

$$\begin{aligned} \text{DSPACE}(n^k) &\subset \text{DSPACE}(\log(n) \cdot n^k) \\ &\subset \text{DSPACE}(\log(n)^{l+1} \cdot n^k) \\ &\subset \text{DSPACE}(\sqrt{n} \cdot n^k) \\ &\subset \text{DSPACE}(n^{k+1}) \\ &\subset \text{DSPACE}(2^n) \\ &\subset \text{DSPACE}(\log(n) \cdot 2^n) \\ &\subset \text{DSPACE}(\sqrt{n} \cdot 2^n) \\ &\subset \text{DSPACE}(n \cdot 2^n) \\ &\subset \text{DSPACE}(2^{(l+1)n}) \end{aligned}$$

Weiter gibt es zu jedem rekursiven $s : \mathbb{N} \rightarrow \mathbb{N}$ nach Lemma 7.2 ein platzkonstruierbares S mit $S(n) \geq n \cdot s(n)$. Nach dem Hierarchiesatz gilt, dann $\text{DSPACE}(s(n)) \subset \text{DSPACE}(S(n))$. Es gibt also keine maximale deterministische Platzklasse. (Wir haben dies bereits allgemein für beliebige abstrakte Komplexitätsklassen im letzten Abschnitt beobachtet.) \diamond

BEWEIS VON SATZ 7.3. Wir geben einen $S(n)$ -platzbeschränkten deterministischen off-line Mehrband-Turingakzeptor M an, so dass die von M erkannte Sprache $L = L(M)$ nicht in $\text{DSPACE}(s(n))$ liegt.

Um $L \notin \text{DSPACE}(s(n))$ zu sichern, diagonalisiert M gegen alle $s(n)$ -platzbeschränkten Maschinen mit Hilfe des off-line Akzeptors U_1 , der universell für die deterministischen 1-Band-off-line Akzeptoren ist (selbst also 3 Arbeitsbänder hat). M selbst verfügt über 5 Arbeitsbänder, wobei es auf den ersten drei Bändern U_1 simuliert, Band 4 als Eingabeband für U_1 mitverwendet (neben dem tatsächlichen Eingabeband), und Band 5 für einen Schrittzähler benutzt. Die M -Rechnungen zerlegen wir in zwei Phasen, die Initialisierungs- und die Simulationsphase.

In der Initialisierungsphase werden Vorkehrungen getroffen, dass M total ist und der Platzschranke $S(n)$ genügt. Ist die Eingabe w ein kodiertes Paar $\langle x, y \rangle$, so simuliert M in der Simulationsphase U_1 für Eingabe (x, w) . Kann die Simulation beendet werden, so diagonalisiert M gegen die normierte Maschine M_x , indem es w genau dann akzeptiert, wenn U_1 die Eingabe (x, w) - also M_x die Eingabe w - verwirft (vgl. die Definition von U_1). Auf diese Art wird sichergestellt werden, dass $L(M) \neq L(M_x)$ für jeden normierten $O(s(n))$ -platzbeschränkten off-line 1-Band-Turingakzeptor M_x gilt, weshalb - wegen des Bandreduktionssatzes und des Normierungslemmas - in der Tat $L(M) \neq L(M')$ für alle $s(n)$ -platzbeschränkten off-line Mehrband-Turingakzeptoren M' gilt und damit $L(M) \notin \text{DSPACE}(s(n))$.

Wir beschreiben die Arbeitsweise von M bei Eingabe w , $|w| = n$, nun etwas genauer.

In der Initialisierungsphase markiert M auf den Bändern 1-5 die Felder mit Adressen aus $[-S(n), +S(n)]$. Wegen der Platzkonstruierbarkeit von $S(n)$ ist dies ohne zusätzlichen Platzbedarf (und wegen des Bandreduktionssatzes ohne Hinzunahme zusätzlicher Bänder) möglich. Auf Band 5 wird ein Binärzähler der Länge $2S(n) + 1$ initialisiert, d.h. die dort markierten Felder werden mit Nullen beschriftet. Wird in der Simulationsphase der markierte Bereich auf einem Band verlassen, so stoppt M sofort und verwirft. Insbesondere gilt dies auch, wenn der Zähler auf Band 5 überläuft.

In der Simulationsphase zerlegt M zunächst die Eingabe w in $w = 1^{|x|}0xy$ und kopiert x auf Band 4. (Stellt sich hierbei heraus, dass w solch eine Zerlegung nicht erlaubt oder der markierte Platz auf Band 4 zur Aufnahme von x nicht ausreicht, so stoppt M und verwirft.) M beginnt dann U_1 Schritt-für-Schritt für die Eingabe (x, w) zu simulieren. (Hierbei übernimmt Band 4 z.T. die Rolle des Eingabebandes, nämlich für die erste Eingabe x .) Bei jedem simulierten U_1 -Schritt wird der Zähler auf Band 5 inkrementiert. (Da bei Überlauf des Zählers die Rechnung abgebrochen wird, sichert dies die Terminierung von M im Falle, dass U_1 nicht terminiert.) Kann die Simulation beendet werden, so akzeptiert M die Eingabe w genau dann, wenn U_1 seine Eingabe (x, w) verwirft. (Hier ist es wesentlich, dass wir es mit deterministischen Maschinen zu tun haben, d.h. das Akzeptanzverhalten hier umkehren können, indem wir akzeptierende und verworfende Stoppzustände gerade vertauschen, was keine zusätzlichen Ressourcen erfordert).

Die getroffenen Vorkehrungen sichern offensichtlich, dass M total (Zähler!) und $S(n)$ -platzbeschränkt ist, also

$$L \in \text{DSPACE}(S(n))$$

gilt. Zum Nachweis, dass $L \notin \text{DSPACE}(s(n))$ gilt, gehen wir vom Gegenteil aus, um hieraus einen Widerspruch abzuleiten.

Die Annahme $L \in \text{DSPACE}(s(n))$ impliziert, dass L von einer totalen $s(n)$ -platzbeschränkten off-line Turingmaschine M' erkannt wird. Wegen des Bandreduktionssatzes und des Normierungslemmas gibt es dann aber auch einen normierten $O(s(n))$ -platzbeschränkten off-line 1-Band-Turingakzeptor M_x mit $L = L(M_x)$. Nach Definition von U_1 gilt also für alle Wörter w

$$\begin{aligned} w \in L &\Leftrightarrow M_x \text{ akzeptiert } w \text{ (mit Platzbedarf } O(s(|w|))) \\ &\Leftrightarrow U_1 \text{ akzeptiert } (x, w) \text{ (mit Platzbedarf } O(s(|w|))) \end{aligned}$$

Es folgt mit Lemma 3.9 (und $\log(n) \leq s(n)$), dass die Anzahl der erreichbaren Konfigurationen von U_1 bei Eingabe (x, w) durch $2^{O(s(|w|))}$ beschränkt ist. Da im deterministischen Fall endliche Rechnungen schleifenfrei sind, d.h. keine Konfigurationswiederholungen enthalten, gibt es daher eine Konstante $c \geq 1$, sodass

$$\begin{aligned} w \in L &\Leftrightarrow U_1 \text{ akzeptiert } (x, w) \text{ mit Platzbedarf } \leq c \cdot s(|w|) \\ &\text{in Zeit } \leq 2^{c \cdot s(|w|)} \end{aligned} \quad (7.1)$$

für alle Wörter w der Länge $\geq c$ gilt. Da es wegen (S1) unendlich viele Zahlen n mit

$$S(n) > c \cdot s(n)$$

gibt, können wir solch ein n wählen, sodass zusätzlich $n \geq c$, $n > 2|x| + 1$ und $|x| \leq \log(n) (\leq S(n))$ gelten. (Diese zusätzlichen Bedingungen werden von fast allen n erfüllt!) Sei nun w ein Wort dieser Länge n , das die Gestalt $w = 1^{|x|}0xy$ hat (z.B. $w = 1^{|x|}0x0^{n-(2|x|+1)}$). Nach (7.1) und Wahl von n simuliert dann M bei Eingabe w die Maschine U_1 bei Eingabe (x, w) vollständig, d.h.

$$w \in L(M) \Leftrightarrow U_1 \text{ verwirft } (x, w) \Leftrightarrow w \notin L$$

Da aber L als $L(M)$ definiert war, ist dies unmöglich. \square

Zum Beweis eines Zeithierarchiesatzes geht man entsprechend vor. Da hier jedoch – wegen des Zeitverlustes bei der Bandreduktion – die bekannten universellen Maschinen nicht ohne Zeitverlust auskommen, erhalten wir gewisse Lücken.

7.5 SATZ. (DETERMINISTISCHER ZEITHIERARCHIESATZ) Seien $t(n), T(n) > n$ rekursive Funktionen, für die

(T1) $T(n) \notin O(t(n) \log(t(n)))$ und

(T2) $T(n)$ zeitkonstruierbar

gilt. Dann gilt $\text{DTIME}(T(n)) \not\subseteq \text{DTIME}(t(n))$. Sind t und T so gewählt, dass zusätzlich $t(n) \leq T(n)$ fast überall gilt, so gilt also

$$\text{DTIME}(t(n)) \subsetneq \text{DTIME}(T(n)).$$

BEWEISIDEE. Ähnlich wie im Beweis des Platzhierarchiesatzes geben wir einen deterministischen $O(T(n))$ -zeitbeschränkten on-line Mehrbandakzeptor M an, so dass die von M erkannte Sprache $L = L(M)$ nicht in $\text{DTIME}(t(n))$ liegt. Wegen der linearen Beschleunigung gilt dann $L \in \text{DTIME}(T(n)) - \text{DTIME}(t(n))$.

Die Maschine M simuliert nun den on-line Akzeptor U_2 , der universell für die det. 2-Band-on-line-Akzeptoren ist. Die Initialisierungsphase entfällt hier. Bei Eingabe $w = 1^{|x|}0xy$ simuliert M den universellen Akzeptor U_2 für Eingabe (x, w) wiederum Schrittfür-Schritt und akzeptiert genau dann, wenn U_2 verwirft. Um sicherzustellen, dass M $T(n)$ -zeitbeschränkt ist, zählt man die M -Schritte und bricht die Simulation nach $T(n)$ -Schritten im verwerfenden Zustand ab. Da T zeitkonstruierbar ist, lässt sich der Zähler

dadurch realisieren, dass man eine Maschine M_T mit Laufzeit $T(|w|)$ parallel mitlaufen lässt.

Der Nachweis, dass $L \notin \text{DTIME}(t(n))$ gilt, wird wiederum indirekt geführt. Aus der Widerspruchsannahme lässt sich hier durch Reduktion auf 2 Bänder und Normierung ein Index x gewinnen, sodass

$$w \in L \Leftrightarrow U_2 \text{ akzeptiert } (x, w) \text{ in Zeit } O(t(|w|) \cdot \log(t(|w|)))$$

für alle w gilt. Wegen der (stärkeren) Annahme (T1) reicht dies aus, um – ähnlich wie Platzhierarchiesatz – ein w mit

$$w \in L \Leftrightarrow w \notin L(M)$$

im Widerspruch zu $L = L(M)$ zu finden. \square

7.6 BEISPIEL. Die im Anwendungsbeispiel zum Platzhierarchiesatz gezeigten echten Inklusionen für Platzklassen können wir aus dem Zeithierarchiesatz nur z.T. für die entsprechenden Zeitklassen herleiten. So gilt weiterhin (für $k, l \geq 1$)

$$\begin{aligned} \text{DTIME}(n^k) &\subset \text{DTIME}(\log(n)^{l+1} \cdot n^k) \\ &\subset \text{DTIME}(\sqrt{n} \cdot n^k) \\ &\subset \text{DTIME}(n^{k+1}) \\ &\subset \text{DTIME}(2^n) \\ &\subset \text{DTIME}(2^{(l+1)n}). \end{aligned}$$

Ob jedoch z.B. die Inklusionen

$$\text{DTIME}(n^k) \subseteq \text{DTIME}(\log(n) \cdot n^k)$$

und

$$\text{DTIME}(2^n) \subseteq \text{DTIME}(n \cdot 2^n) \quad (7.2)$$

ebenfalls echt sind, lässt sich mit Hilfe des Zeithierarchiesatzes nicht entscheiden, da

$$\lim_{n \rightarrow \infty} \frac{n^k \cdot \log(n^k)}{\log(n) \cdot n^k} = k$$

und

$$\lim_{n \rightarrow \infty} \frac{2^n \cdot \log(2^n)}{n \cdot 2^n} = 1$$

gilt. \diamond

Wegen der Lücken im Zeithierarchiesatz hat man andere Techniken zur Trennung von Zeitklassen eingeführt. Eine recht einfache Methode sind hierbei Translationslemmata, von denen wir hier ein Beispiel angeben.

7.7 SATZ. (TRANSLATIONSLEMMA) Seien $t(n), T(n), f(n) > n$ schwach monotone zeitkonstruierbare Funktionen, wobei $f(n)$ hyperlinear sei. Gilt dann

$$\text{DTIME}(t(n)) \subseteq \text{DTIME}(T(n)), \quad (7.3)$$

so gilt auch

$$\text{DTIME}(t(f(n))) \subseteq \text{DTIME}(T(f(n))). \quad (7.4)$$

Bevor wir das Translationslemma beweisen, zeigen wir, wie wir hieraus die Echtheit der Inklusion in (7.2) herleiten können. Wir benutzen dabei, dass eine angenommene Lücke in der Zeithierarchie durch die Funktion f an andere Stelle übertragen und dort *aufgebläht* wird. Bei geeigneter Wahl von f wird die neue Lücke dadurch so groß, dass sie dem Hierarchiesatz widerspricht.

7.8 KOROLLAR. $\text{DTIME}(2^n) \subset \text{DTIME}(n \cdot 2^n)$.

BEWEIS. Für einen Widerspruch nehmen wir $\text{DTIME}(2^n) = \text{DTIME}(n \cdot 2^n)$ an. Dann gilt insbesondere $\text{DTIME}(n \cdot 2^n) \subseteq \text{DTIME}(2^n)$. Wenden wir das Translationslemma auf $t(n) = n \cdot 2^n$, $T(n) = 2^n$ und auf die beiden Funktionen $f(n) = n + 2^n$ bzw. $f(n) = 2^n$ an, so folgt hiermit:

$$\text{DTIME}((n + 2^n) \cdot 2^{(n+2^n)}) \subseteq \text{DTIME}(2^{(n+2^n)}) \quad (7.5)$$

bzw.

$$\text{DTIME}(2^n \cdot 2^{2^n}) \subseteq \text{DTIME}(2^{2^n}). \quad (7.6)$$

Da $2^{(n+2^n)} = 2^n \cdot 2^{2^n}$, ergeben (7.5) und (7.6) zusammen

$$\text{DTIME}((n + 2^n) \cdot 2^{(n+2^n)}) \subseteq \text{DTIME}(2^{2^n}). \quad (7.7)$$

Für $T'(n) = (n + 2^n) \cdot 2^{(n+2^n)}$ und $t'(n) = 2^{2^n}$ gilt jedoch $T'(n) \notin O(t'(n) \cdot \log(t'(n)))$, da wegen

$$\frac{t'(n) \cdot \log(t'(n))}{T'(n)} = \frac{2^{2^n} \cdot \log(2^{2^n})}{(n + 2^n) \cdot 2^{(n+2^n)}} \leq \frac{2^{2^n} \cdot 2^n}{(n + 2^n) \cdot 2^n \cdot 2^{2^n}} = \frac{1}{n + 2^n}$$

dieser Quotient gegen 0 konvergiert. Da t' und T' zeitkonstruierbar sind, gilt also nach dem Zeithierarchiesatz

$$\text{DTIME}(2^{2^n}) \subset \text{DTIME}((n + 2^n) \cdot 2^{(n+2^n)})$$

im Widerspruch zu (7.7). □

BEWEIS VON SATZ 7.7. Wir nehmen (7.3) an und wollen (7.4) zeigen. Es genügt hierzu für eine gegebene Sprache $L \in \text{DTIME}(t(f(n)))$ zu zeigen, dass L auch in $\text{DTIME}(T(f(n)))$ liegt. Hierzu definieren wir zunächst die Sprache

$$L_f = \{x_f : x \in L\}$$

wobei

$$x_f = 1^{f(|x|) - (|x|+1)} 0x$$

(da f hyperlinear ist, können wir o.B.d.A. $f(n) > n + 1$ für alle n annehmen).

D.h. L_f ist eine aufgeblähte (padded) Version von L , bei der die Länge der Eingaben in kanonischer Weise um den Faktor f aufgebläht sind.

Da f zeitkonstruierbar ist, kann man in $O(|y|)$ Schritten prüfen, ob $y = x_f$ für ein Wort x gilt, und gegebenenfalls dieses x berechnen. Aus

$$L \in \text{DTIME}(t(f(n)))$$

folgt hiermit aber (mit linearer Beschleunigung), dass

$$L_f \in \text{DTIME}(t(n))$$

gilt. Um $y \in L_f$ zu entscheiden, berechnet man nämlich zunächst in $O(n) \leq O(t(n))$ Schritten das Wort x mit $y = x_f$ (falls existent) und entscheidet dann in $O(t(f(|x|))) = O(t(|y|))$ Schritten, ob $x \in L -$ und damit $x_f \in L_f -$ gilt. Nach (7.3) gilt dann aber auch

$$L_f \in \text{DTIME}(T(n)).$$

Hieraus erhalten wir aber $L \in \text{DTIME}(T(f(n)))$, indem wir zur Entscheidung, ob $x \in L$ gilt, in $T(|x_f|) = T(f(|x|))$ Schritten testen, ob $x_f \in L_f$ gilt. \square

Im Beweis der Hierarchiesätze haben wir benutzt, dass die deterministischen Platz- und Zeitklassen (effektiv) gegen Komplement abgeschlossen sind, d.h. dass wir das Ergebnis einer terminierenden Rechnung umkehren können, indem wir akzeptierende und verwerfende Stoppzustände vertauschen. Da die Akzeptanz bei nichtdeterministischen Maschinen nicht lokal definiert ist, liefert das Vertauschen der akzeptierenden und verwerfenden Stoppzuständen bei einer nd. totalen Turingmaschine nicht das Komplement der akzeptierten Sprache. Die Beweise der Hierarchiesätze lassen sich daher auf den nichtdeterministischen Fall nicht direkt übertragen. In der Tat ist es bis heute nicht bekannt, ob die (natürlichen) nd. Zeitklassen gegen Komplement abgeschlossen sind, und man vermutet sogar, dass dies nicht der Fall ist. Für die nd. Platzklassen konnte man dagegen den (effektiven) Abschluss gegen Komplement nachweisen und damit den Beweis des Platzhierarchiesatzes auf den nichtdeterministischen Fall ausweiten. Dieses Ergebnis ist jedoch viel neuer als die Hierarchiesätze, weshalb man sich zuvor andere Methoden zum Beweis für nichtdeterministische Hierarchiesätze überlegt hat. Diese benutzen in der Regel Translationslemmata. Es ist daher interessant zu beobachten, dass das oben vorgestellte Translationslemma für alle hier betrachteten Komplexitätsmaße, auch die nichtdeterministischen, gilt. (Analysiert man den Beweis, so sieht man, dass das Argument auch für die Platzkomplexität durchgeht und wir det. durch nd. Maschinen durchgängig ersetzen können.) Im nächsten Kapitel, in dem wir die nichtdeterministischen Komplexitätsklassen genauer untersuchen werden, werden wir an einem Beispiel zeigen, wie das Translationslemma zur Trennung nd. Platzklassen benutzt werden kann.